Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (*and Subtitle*) DRAMA: A Simplified Spares Optimization Model	5. TYPE OF REPORT & PERIOD COVERED
	6. PERFORMING ORG. REPORT NUMBER LMI Task ML008
7. AUTHOR(s) Richard M. Fabbro; John B. Abell	8. CONTRACT OR GRANT NUMBER(s) MDA 903-77-C-0370
9. PERFORMING ORGANIZATION NAME AND ADDRESS Logistics Management Institute, 4701 Sangamore Road, Washington, D. C. 20016	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Assistant Secretary of Defense (Manpower, Reserve Affairs and Logistics)	12. REPORT DATE February 1980
	13. NUMBER OF PAGES 69
14. MONITORING AGENCY NAME & ADDRESS (*if different from Controlling Office*)	15. SECURITY CLASS. (*of this report*) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (*of this Report*)

"A" Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (*of the abstract entered in Block 20, if different from Report*)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (*Continue on reverse side if necessary and identify by block number*)

Availability; Spares Cost; Spares Optimization

20. ABSTRACT (*Continue on reverse side if necessary and identify by block number*)

This report describes the logic and operation of a simplified spares optimization model called DRAMA. The model has been used to better understand the relationship among availability, reliability, and spares investment levels for new weapons systems. Systems modeled by DRAMA include the LAMPS-Mark III, the SOTAS, and the XM-1.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (*When Data Entered*)

DRAMA: A SIMPLIFIED
SPARES OPTIMIZATION MODEL

ADA Φ84 1Φ3

February 1980

Richard Fabbro
John B. Abell

Prepared pursuant to Department of Defense Contract No. MDA 903-77-C-0370 (Task ML908).

LOGISTICS MANAGEMENT INSTITUTE
4701 Sangamore Road
Washington, D. C. 20016

PREFACE

This report describes the logic and operation of the simplified spares optimization model called DRAMA. The model has been used to understand better the relationships among availability, reliability, and spares investment levels for new weapon systems.

ACKNOWLEDGMENTS

We are indebted to our colleagues at LMI, Michael J. Konvalinka and F. Michael Slay, for several important contributions to this report.

EXECUTIVE SUMMARY

DRAMA is an analytic model that relates the availability of a weapon system to the cost of its spares. It is a model of a multi-item, multi-location, two-echelon supply system in which each item may be stocked at a depot and two or more similar bases. Item demands are assumed to be Poisson-distributed. DRAMA focuses on only one weapon system at a time, so users who are interested in modeling multiple weapon systems are referred to the LMI Aircraft Availability Model, of which DRAMA is a simplified version.

DRAMA produces a set of points comprising an availability-versus-cost curve. Each point is an optimum, i.e., it represents the least-cost mix of spares for that level of availability or, conversely, the greatest availability for that level of spares investment. Availability is defined as the probability that an end item (such as an aircraft or tank) is <u>not</u> waiting for a component to be repaired or be shipped to it; thus, availability is defined from the point of view of the supply system.

The potential user should feel encouraged to use DRAMA for certain applications for which it may not appear, at first glance, to be entirely appropriate. For example, despite its assumption of a two-echelon supply system in which all bases are equal, DRAMA has successfully modeled the Army's five-echelon system, and has represented situations where bases were not equivalent.

In comparison with some spares-optimization models, DRAMA requires relatively little input data. For each component, DRAMA needs removal-rate, echelon-of-repair, cost, and application data. For the end item, it needs

utilization and deployment data and base repair, depot repair, order-and-shipment, and condemnation lead times.

DRAMA's primary output is a table of spares costs at various levels of availability. Optional outputs, including quantities of items and costs, are also available.

TABLE OF CONTENTS

I. THE ALGEBRAIC MODEL

THE INVENTORY SYSTEM

DRAMA[1] is a model of a multi-item, multi-location, two-echelon inventory system consisting of two or more identical using locations called bases and a single, central resupply location called a depot. Each base is equipped with end items, such as aircraft or tanks. The end items are comprised of various components critical to their operation. These components are usually called Line Replaceable Units (LRUs) and are to be distinguished from Shop Replaceable Units (SRUs) which are subcomponents of LRUs and which are not directly removable from end items. DRAMA models only LRUs, not SRUs.

The purpose of DRAMA is to maximize the availability of end items subject to a constraint on the total cost of spare components or, conversely, to find the least cost mix of spare components that will result in a specified availability. Availability is defined as the probability that an end item selected at random will not be waiting for a component to be repaired or be shipped to it.

When a component fails, it is removed from the end item and sent to base repair. If a serviceable spare component is available, it is installed on the end item. (By serviceable we mean in working condition, suitable for use in the end item.) The base repair shop decides whether the repair of the failed component is within its capability. If it is, it repairs the component and gives it to base supply as a serviceable spare.

---

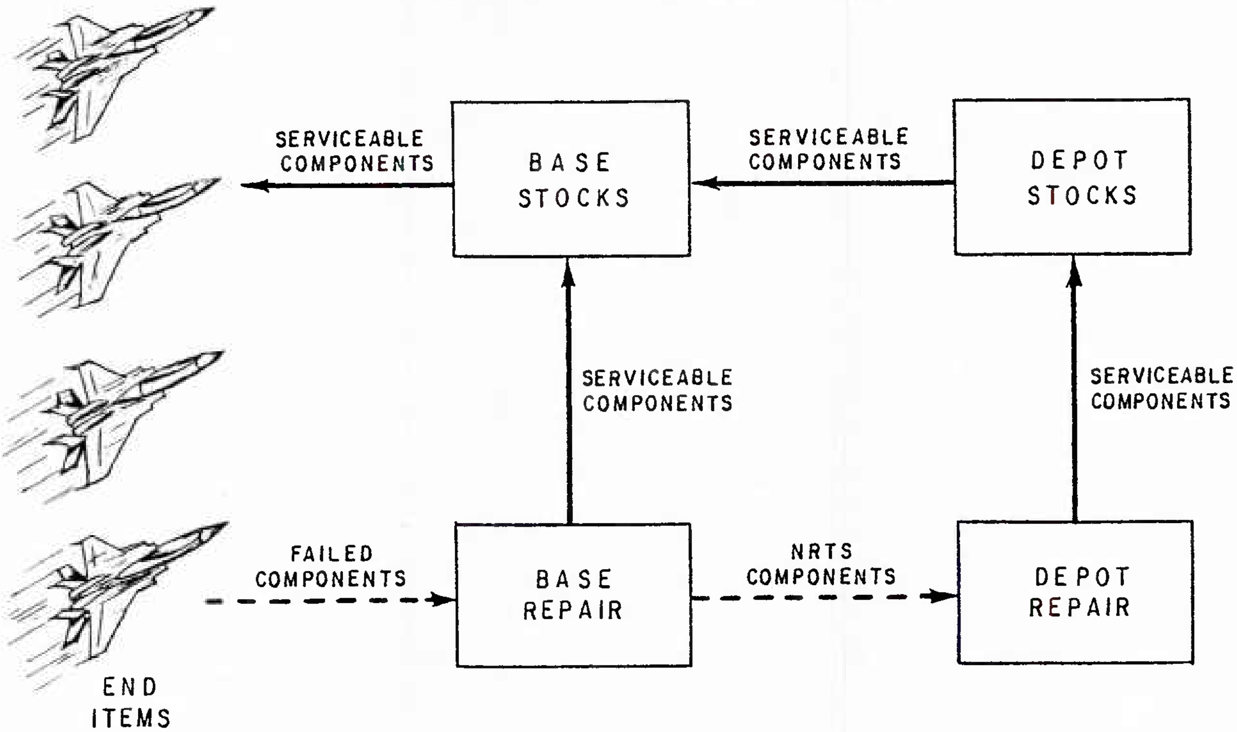
[1] DRAMA is an acronym for Diagnostic Reliability, Availability, and Modularity Analyzer. The name comes from its original application, which was to assess the impact of modularization and fault-diagnosis reliability on spares costs.

If the repair of the failed component is beyond the repair capability of the base, it is declared to be not-repairable-this-station (NRTS) and shipped to depot repair. In this event, a replacement item is immediately ordered by the base from the depot. Thus, the inventory system operates according to an (S-1,S) inventory policy, i.e., whenever the number of components on-hand at the base plus due-in from the depot minus due-out to the operational organization falls below the authorized stock level, S, another component is ordered immediately.

The flow of components in the system is described in Figure I-1.

FIGURE I-1

FLOW OF COMPONENTS



The average time that elapses from the removal of a failed component from an end item until it is returned to serviceable condition (given that it is

not NRTS) by the base repair shop is called the base repair time. The components in the process of repair at the base are said to be in the base repair pipeline. We define a pipeline as an expected value, e.g., the base repair pipeline of a component is the expected number of that component in repair at the base.

Similarly, the average time that elapses from the removal of a failed component from an end item (in the case where the component is NRTS) until it is returned to serviceable condition by the depot is called the depot repair time. Note that the depot repair time includes the time required to ship the failed component from the base to the depot. That shipment is referred to as retrograde shipment. Thus, the depot repair pipeline includes the retrograde pipeline.

The average time that elapses from the ordering of a serviceable component by the base from the depot until the component is received by the base is called the order-and-ship time. We assume that the determination of NRTS status and the placement of the order are instantaneous.

Some proportion of the NRTS components sent to the depot for repair may also be beyond the repair capability of the depot, or the cost of repair may exceed the cost of a replacement item. In this case, the depot condemns the component and orders a replacement from the manufacturer. The average time that elapses from the removal of such a component from an end item until it is received by the depot from the manufacturer is called the condemnation lead time or procurement lead time. This lead time is combined with the depot repair time for our purposes so that whenever we refer to the depot repair pipeline, we include the condemnation pipeline.

Any component that is in the base repair, depot repair (including retrograde), or order-and-ship pipeline is said to be in resupply. The resupply

pipeline (or simply pipeline) is the sum of these three component pipelines. The average time a component is in the resupply pipeline is called the resupply time.

The inventory of spare components is positioned at the depot as depot stock and at the bases as base stock. Whenever an order for a serviceable component is received by a base and the base is out of stock, a base backorder occurs. Similarly, when an order is received by the depot and the depot has no stock, a depot backorder occurs. The time that elapses from the receipt of an order by the depot until a serviceable component is available at the depot is called the depot delay time. If one or more serviceable spares are in depot stock when the order is received, the depot delay time is zero.

THE MODEL

The fundamental purpose of DRAMA is to produce a set of points describing the relationship between end-item availability and the initial investment cost for spare components for depot and base stocks. Each point represents the least-cost mix of spares for its corresponding availability. In other words, DRAMA produces a spares list that maximizes end-item availability given a budget constraint; conversely, it minimizes spares costs for any specified level of end-item availability.

We will now show how DRAMA models this logistics system. We assume that demands for components are equivalent to removals of components due to failures and that: (1) demands follow a Poisson distribution and (2) demands for any component are independent of demands for any other component. Then, we define for any particular component:

T = average resupply time,

t_b = average base repair time,

t_d = average depot repair time,

t_o = average order-and-ship time,

t_w = average depot delay time,

t_c = average condemnation lead time,

ϕ = fraction of NRTS components condemned by the depot,

r = base repair fraction (i.e., 1 - NRTS rate),

λ = demand rate,

X = a random variable: the number of components in resupply, and

$B(S_o, S)$ = a random variable: the number of base-level backorders given a depot stock level of S_o and a base stock level of S.

In accordance with our Poisson assumption, the probability density function (p.d.f.) of X is given by

$$p(x) = \begin{cases} \dfrac{e^{-\lambda T} (\lambda T)^x}{x!}, & x = 0, 1, 2, \ldots \\ 0 \text{ elsewhere} \end{cases}$$

where T is a weighted combination of pipeline times given by

$$T = r\, t_b + (1 - r)(t_o + t_w).$$

Despite the fact that T is, in fact, a random variable, we may use it here like a constant due to a theorem and proof of C. Palm.[2] Treating T as a constant, we can express the expected number of components in resupply as

$$E(X) = \lambda T.$$

The expected number of base-level backorders (EBO), given a base stock level of S, and a depot stock level of S_o, is simply

$$E[\ B(S_o,S)\ ] = \sum_{x>S} p(x)\ (x - S).$$

---

[2]Palm, C.,"Analysis of the Erlang Traffic Formula for Busy-Signal Arrangements", Ericsson Technics, No.5, 1938, pp.39-58.

The depot delay time is a function of the depot stock level and the depot demand rate. The greater the depot stock level, the less the average waiting time for a serviceable spare. Since we assume a steady-state system, the expected depot delay time is equal to the product of the expected number of depot-level backorders and the mean time between depot demands, i.e.,

$$t_w = \frac{E[D(S_o)]}{\lambda(1-r)}$$

where D, a random variable, denotes the number of depot-level backorders and λ (1-r) is the depot demand rate. We define N as the number of components in the depot repair pipeline (including the condemnation pipeline). By Palm's Theorem, N has a Poisson distribution. Its expectation is equal to

$$\Delta = \lambda\ (1 - r)\ [\phi t_c + (1 - \phi))t_d]$$

and its p.d.f. is simply

$$g(n) = \begin{cases} \dfrac{e^{-\Delta}\Delta^n}{n!}, & n = 0, 1, 2, \ldots \\ 0 \text{ elsewhere}. \end{cases}$$

Finally, $E[D(S_o)]$, the expected number of depot-level backorders is computed as

$$E[D(S_o)] = \sum_{n>S_o} g(n)\ (n-S_o).$$

We will now extend the model in a straightforward manner to the multi-item, multi-base system. In our notation, we will let i denote the ith component type where there are n different component types and we will assume that there are k identical bases. Then, for component i,

X_i = a random variable: the number of components of type i in resupply,

T_i = resupply time,

$t_{w,i}$ = depot delay time,

ϕ_i = fraction of NRTS components condemned by the depot,

r_i = base repair fraction,

λ_i = demand rate at each base,

$p_i(x)$ = the p.d.f. of X_i,

$S_{o,i}$ = depot stock level,

S_i = stock level at each base,

$B_i(S_{o,i},S_i)$ = a random variable: base-level backorders at each base,

$D_i(S_{o,i})$ = a random variable: depot-level backorders,

N_i = a random variable: number of components of type i in the depot repair pipeline, and

$g_i(n)$ = the p.d.f. of N_i.

The variables t_b, t_d, t_o, and t_c are defined as before; they are assumed to apply to all components.

Thus,

$$E[D_i(S_{o,i})] = \sum_{n>S_{o,i}} g_i(n)(n-S_{o,i})$$

and $E(N_i) = k\,\lambda_i\,(1-r_i)\ [\phi_i t_c + (1-\phi_i)t_d]$.

Denote $E(N_i)$ by Δ_i. Then,

$$g_i(n) = \begin{cases} \dfrac{e^{-\Delta_i}\,\Delta_i^{\,n}}{n!}, & n = 0,\ 1,\ 2,\ \ldots \\ 0 \text{ elsewhere}. \end{cases}$$

The depot delay time for component i is

$$t_{w,i} = \frac{E[D_i(S_{o,i})]}{k\lambda_i(1-r_i)}\ .$$

The p.d.f. of X_i becomes simply

$$p_i(x) = \begin{cases} \dfrac{e^{-\lambda_i T_i} (\lambda_i T_i)^x}{x!} \\ \\ 0 \text{ elsewhere} \end{cases}$$

where $T_i = r_i t_b + (1 - r_i)(t_o + t_{w,i})$; so

$$E[B_i(S_{o,i}, S_i)] = \sum_{x > S_i} p_i(n)(n - S_i).$$

THE CONCEPT OF AVAILABILITY

We will now consider the relationship between base-level backorders and end-item availability. Suppose component i has a quantity per end item of q_i, that there are u end items at each base, and that there are k bases and n different component types in the system, as before.

An estimate of the probability that component i is missing from any one of its positions in an end item is

$$\frac{E[B_i(S_{o,i}, S_i)]}{uq_i} .$$

The estimated probability it is not missing, then, is

$$1 - \frac{E[B_i(S_{o,i}, S_i)]}{uq_i}$$

and that it is not missing in any of its positions on an end item is

$$\left[1 - \frac{E[B_i(S_{o,i}, S_i)]}{uq_i} \right]^{q_i} .$$

Finally, the estimated probability that no component is missing on an end item is

$$A = \prod_{i=1}^{n} \left[1 - \frac{E[B_i(S_{o,i}, S_i)]}{uq_i} \right]^{q_i} .$$

"A" denotes the end-item availability rate. It is this function that DRAMA maximizes subject to a constraint on spares investment.

Maximizing A is equivalent to maximizing the logarithm of A; indeed, that is the objective function operated on by DRAMA. This objective function is separable; i.e., it allows us to compute the ratio of marginal availability per unit cost for each component type independent of all other component types. The actual solution technique employs marginal analysis. It proceeds in an iterative fashion. At each iteration it computes the end-item availability that would result from an incremental addition of spare components to the system for each component type. It then selects the component type that results in the greatest increase in end-item availability per unit cost. These computations are based on the allocation of spare components among the depot and all the bases that yields the minimal number of expected base-level backorders in the system. The solution technique is explained further in Chapter II.

Three fundamentally important assumptions underlie DRAMA's computations. The first is that component demands are independent of one another. The second is that the process that generates demands (e.g., a flying hour or sortie program for aircraft or a number of miles of operation for tanks) is known and does not change as a function of spares levels. The third assumption is that the system is in a steady state, i.e., the demand process is stationary.

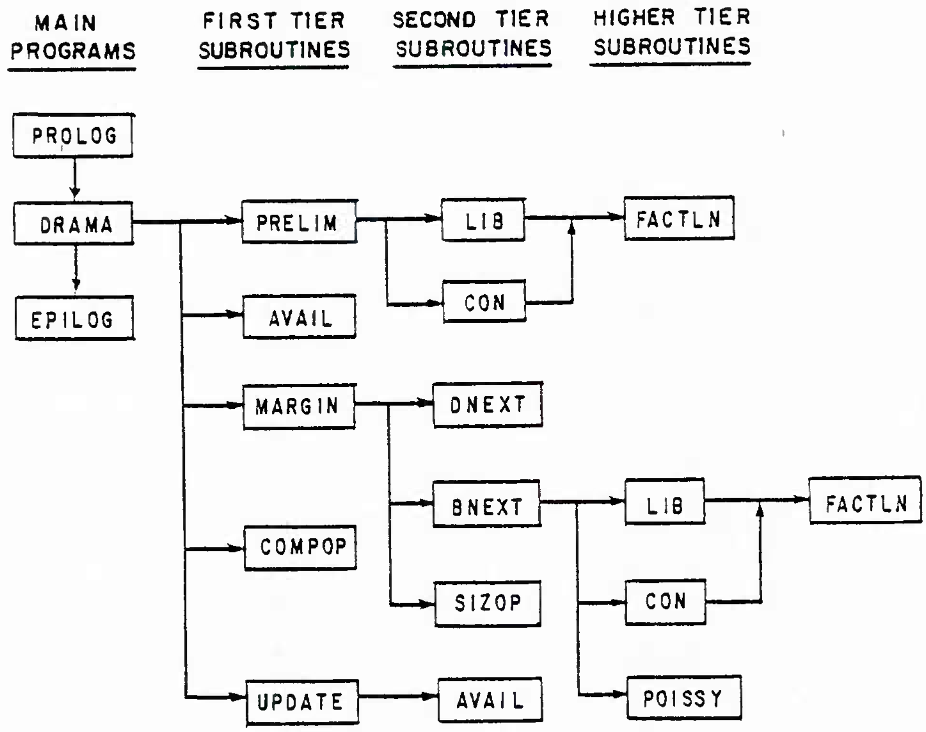
II. THE COMPUTER IMPLEMENTATION

STRUCTURE

To economize on core, the input and output functions of DRAMA have been delegated to separate programs. These programs are respectively named PROLOG and EPILOG, and are discussed in Chapter III. This chapter focuses exclusively on the main program, DRAMA, and its subroutines.

DRAMA consists of a main program and twelve subroutines. The logical links between subroutines are shown in Figure II-1. The functions of each

FIGURE II-1
DRAMA SUBROUTINE STRUCTURE



subroutine are discussed briefly below, and are more fully described in the remaining sections of this chapter. We now define the critical variables that the subroutines employ. These variables are:

BEBO: The expected number of backorders of a component at each base. It is a function of the spares inventories at the depot and at each base. In Chapter I, it was defined as $B[S_0,S]$.

DEBO: The expected number of backorders of a component at the depot. It is a function of the spares inventory at the depot, and it was defined in Chapter I as $E[D(S_0)]$.

BMEAN: The mean value of the Poisson distribution used in the computation of BEBO. BMEAN is equal to the expected number of backorders that a base would have if it had no spares, $E[B(S_0, 0)]$. Initially, when neither the bases nor the depot have spares, BMEAN is also equal to the expected number of parts in resupply to each base. This initial value of BMEAN is a distinct variable in DRAMA called BMEANØ. In Chapter I's notation, BMEAN is equivalent to λT.

DMEAN: The mean value of the Poisson distribution used in the computation of DEBO. It is equal to the expected number of parts in the depot pipeline, where the depot pipeline is understood to include both the <u>condemnation pipeline</u> and the <u>retrograde pipeline</u>, in addition to the <u>depot repair pipeline</u>. DMEAN is equal to the expected number of backorders, DEBO, when the depot has no spares. In Chapter I's notation, DMEAN is equivalent to Δ. Unlike BMEAN, it does not change in value during a DRAMA run.

The subroutine functions can be summarized as follows:

(1) PRELIM: This is a preliminary routine, not involved at all in marginal analysis. At the user's option, it will "purchase" a reasonable, but not optimal, quantity of each component for the depot and for each base.

(2) AVAIL: This subroutine computes component and end-item availabilities.

(3) MARGIN: This routine computes a marginal benefit-to-cost ratio for each component. In the process of computing these ratios, MARGIN determines the purchase quantity that yields the maximum ratio, and it computes the optimal base/depot distribution of the quantity of spares that would be available after the purchase.

(4) COMPOP: This subroutine ranks the components according to their benefit-to-cost ratios. It also selects the component with the highest ratio when a purchase has to be made.

(5) UPDATE: This subroutine recomputes component inventories, system spares costs, and component and end-item availabilities to account for a spares purchase.

(6) LIB: Computes expected backorders at the depot (DEBO) when the depot has a quantity of spares equal to the depot pipeline (DMEAN). Alternatively, it computes expected backorders at each base (BEBO) when each base has a quantity of spares equal to BMEAN.

(7) CON: Computes DEBO given a quantity of depot spares equal to DMEAN minus six standard deviations (a standard deviation equals the square root of the mean). Alternatively, it computes BEBO given a quantity of spares at each base equal to BMEAN minus six standard deviations.

(8) DNEXT: Computes the DEBOs that a component could have after its next spares purchase.

(9) BNEXT: Computes the BEBOs that a component could have after its next spares purchase.

(10) SIZOP: Determines the purchase-size that will yield the greatest benefit for its cost.

(11) FACTLN: Computes the logarithm of the probability that exactly N units will be in the depot pipeline. Alternatively, it computes the logarithm of the probability that exactly N units would be back-ordered at each base if it had no spares.

(12) POISSY: This subroutine computes the value of BEBO that would result from adding K spare units to each base.

WALKTHROUGH

DRAMA, the main program, initializes all of the variables involved in the optimization process. For each component, it sets the counter of depot spares (JSPARE) and the counter of base spares (KSPARE) to zero. In addition, it sets BEBO and DEBO equal to BMEANØ and DMEAN, and it computes values for all of the variables that are later involved in the recomputation of BEBO and DEBO. These variables are discussed extensively in the last section of this chapter. Finally, DRAMA sets the user-option variables (described in Chapter III) to their user-chosen values.

One of the options at the user's disposal is the option of making limited purchases of spares prior to marginal analysis. This is done by calling PRELIM. PRELIM is a particularly useful option for the user who is interested in observing cost-availability relationships at only high levels of availability, for it considerably shortens the total running time of the model without impairing its accuracy. However, for the user who is interested in

observing low levels of availability, PRELIM is less desirable. Because it does not buy "optimal" quantities of each component, PRELIM may result in excessive costs at low availability levels (at higher levels of availability, marginal analysis purchases will have corrected for these suboptimalities) and because running-time problems are less severe at these levels, the advantages of PRELIM are less significant.

If the user wishes to exercise his option of making a preliminary buy, he must assign an appropriate value to the user-option variable IPRE. By setting IPRE equal to two the user instructs PRELIM to buy liberally, and by setting IPRE to one he instructs the subroutine to buy conservatively. Setting IPRE to zero causes DRAMA to bypass PRELIM entirely and to proceed directly to marginal analysis.

When instructed to buy liberally, PRELIM buys a depot-pipeline's worth of spares for the depot. Then, after recomputing the expected number of depot backorders (DEBO) and the Poisson-distribution mean at each base (BMEAN), it buys BMEAN spares for each base. When the quantities DMEAN and BMEAN are not integers, PRELIM buys spares quantities equal to the truncated values of DMEAN and BMEAN. To reinitialize the variables associated with the expected back-order computations, PRELIM must call the subroutine LIB. This process is carried out for each component in the end item.

When instructed to buy conservatively, PRELIM buys a quantity of spares for the depot equal to DMEAN minus six standard deviations. (Note: the standard deviation equals the square root of the mean). Then, after recomputing the values of DEBO and BMEAN, it buys a quantity of spares for each base equal to BMEAN minus six standard deviations. To update the values of the variables used in the expected backorder computations, it calls the subroutine CON. As in the liberal buy, PRELIM truncates spares purchase quantities

whenever BMEAN and DMEAN are non-integral, and it performs its computations independently for each component.

Whether or not the user has exercised his option of making a preliminary buy, the next step in DRAMA is to initialize the availability of every component. This is accomplished by calling the subroutine AVAIL. After the availability initialization, DRAMA begins marginal analysis.

DRAMA's first step in marginal analysis is to compute marginal benefit-to-cost ratios, or ranking values, for each component. This is accomplished by the subroutine MARGIN. The second step is to rank the components according to the magnitude of their ranking values, and to identify the component with the maximum value. This is the task of the subroutine COMPOP. The third step is to "buy" spares of the component with the maximum value. This is accomplished by the subroutine UPDATE. Finally, it is necessary to see whether the cumulative spares cost or the system availability has reached a constraint. If either of the constraints has been reached, marginal analysis will terminate. If not, marginal analysis will continue, and the model will return to step one.

In the second and subsequent iterations of marginal analysis, it is unnecessary to recompute every component's ranking value. This is because the purchase of one type of component has no effect whatsoever on the ranking values of the rest.[1] Therefore, when MARGIN is recalled, it has to recompute only one ranking value, that of the component most recently purchased. Similarly, when COMPOP is recalled, it merely has to find the new ranking position

---

[1]The benefit-to-cost ratios of component purchases are independent of one another because the benefit function in MARGIN is the <u>logarithm</u> of end-item availability. That function, unlike the availability function, is separable by component type.

of the most recently purchased component; the relative positions of the other components remain unchanged.

When marginal analysis ends, DRAMA outputs a set of up to 27 cost-availability points which define the system's optimal cost-availability curve. These points start at an availability level of five per cent and proceed up to an availability of 99 per cent as follows: 5, 10,..., 90, 91, 92,..., 99 per cent. If the availability constraint is less than 99 per cent, or if the cost constraint is exceeded before 99 per cent availability is achieved, then only a partial set of cost-availability points will be output.

To insure that the quantity of spares purchased will be optimal, and to insure that each component has an optimal distribution of spares between the depot and the bases, DRAMA relies on the optimizing logic of MARGIN. MARGIN calculates a component's optimal purchase quantity and spares distribution in the process of computing its benefit-to-cost ratio; then it returns these data to DRAMA so that the component's inventories will be properly updated the next time it's purchased. Because of the importance of MARGIN to the overall process of optimizing inventment in spares across components, the logic of the subroutine is discussed extensively below.

THE LOGIC OF MARGIN

To calculate the optimal purchase quantities and distributions of spares, MARGIN engages in a five-step process. The steps are:

1) Compute the value of DEBO that would result from each purchase alternative. (A purchase alternative is a particular purchase quantity distributed in a particular way. There are a limited number of purchase alternatives for each component.) This is done by DNEXT.

2) Compute the value of BMEAN for each purchase alternative. (BMEAN is a function of DEBO.) This is done at the beginning of BNEXT.

3) Compute the value of BEBO for each purchase alternative. (BEBO is a function of BMEAN and the base spares inventory). This is done in the latter part of BNEXT, and it may involve the subroutines LIB, CON, FACTLN and POISSY.

4) Compute the benefit-to-cost ratio for each purchase alternative (this ratio is a function of BEBO and the purchase quantity involved in the purchase alternative). This is the job of the first part of COMPOP.

5) Find the maximum ratio and identify the purchase alternative that yielded it. This is accomplished in the latter part of COMPOP.

In order to perform these computations in a reasonable amount of time, MARGIN considers only a limited number of purchase alternatives. The number of alternatives considered can never exceed the number of bases being considered plus one (NOB+1). Thus, for a seven-base scenario, there can be no more than eight purchase alternatives. Those alternatives are described below in Table II-1, with j denoting the current spares inventory at the depot, and k denoting the current spares inventory at each base.

TABLE II-1

THE PURCHASE ALTERNATIVES CONSIDERED

FOR A SEVEN-BASE SCENARIO

PURCHASE ALTERNATIVE NUMBER	QUANTITY PURCHASED	RESULTING DEPOT INVENTORY	RESULTING BASE INVENTORY
1	1	j - 6	k + 1
2	2	j - 5	k + 1
3	3	j - 4	k + 1
4	4	j - 3	k + 1
5	5	j - 2	k + 1
6	6	j - 1	k + 1
7	7	j	k + 1
8	1	j + 1	k

Note that MARGIN needs to return only one datum, the purchase-alternative number, to inform DRAMA of the optimal purchase quantity and the optimal spares distribution. The optimal purchase quantity will always equal the purchase-alternative number (NBEST) unless the best purchase-alternative is alternative eight, in which case the purchase quantity will equal one. The resulting depot inventory will always equal the current inventory minus the number of bases plus NBEST (j - NOB + NBEST). The inventory at each base will equal the existing inventory (k) plus one, unless the best purchase alternative is alternative eight. Then the base inventory will remain the same.

Note also that the inventory at the depot can <u>decline</u> even though the total number of spares is increasing. This is because higher availabilities can sometimes be achieved simply by moving spare units from the depot to the bases.

The set of permissible, alternative purchases is constrained by rules designed to minimize an item's expected base-level backorders. Those rules are:

RULE 1: <u>A purchase cannot increase the inventory at any site by more than one unit</u>.

At any site, be it a base or a depot, the EBO reduction of additional spares is strictly decreasing.[2] The second spare reduces EBO less than the first, and the third reduces the EBO less than the second. Therefore, adding one spare to a site will always be more cost-effective than adding two or more.

RULE 2: <u>If a spare is added to the depot, none can be added to the bases, and vice-versa</u>.

Consider the EBO-reduction-to-cost ratios of the following purchases: the purchase of a single depot spare, which reduces

[2]Decreasing marginal EBO is always present, but it will not necessarily be evident until a site's inventory of an LRU exceeds the site's Poisson distribution mean minus six standard deviations.

EBO across all bases by the quantity D; the purchase of N base spares (no more than one per base), where each spare reduces the EBO across all bases by the quantity B; the purchase of one depot spare together with N base spares, where the depot spare reduces EBO by the quantity D, and where <u>each</u> base spare reduces EBO by the quantity B'. Because the EBO-reduction of a spare decreases as BMEAN decreases, and because BMEAN decreases as the depot spares inventory increases, B is greater than B'.

To compute the EBO-reduction-to-cost ratios, we assume for simplicity that each spare costs $1.00. Therefore, the EBO-reduction-to-cost ratios for each purchase are:

D for the purchase of a single depot spare,

B for the purchase of N base spares, and

$\frac{D+NB'}{N+1}$ for the purchase of a depot spare together with N base spares.

Now, if the depot spare is at least as cost-effective as the N base spares, it is more cost-effective than the combined purchase:

$$\begin{aligned} D &\geqq B > B' \\ ND &> NB' \\ (N+1)\,D &> D + NB' \\ D &> \frac{D + NB'}{N + 1} \end{aligned}$$

Similarly, if the N base spares are at least as cost-effective as a single depot spare, then they are more cost-effective than the combined purchase.

$$\begin{aligned} B &\geqq D \\ NB &> NB' \\ (N+1)\,B &> D+NB' \\ B &> \frac{D+NB'}{(N+1)} \end{aligned}$$

Therefore, it will always be more cost-effective to buy <u>either</u> one depot spare <u>or</u> N base spares than to buy a depot spare <u>together with</u> N base spares. As a consequence of this rule, the maximum purchase quantity cannot exceed the number of bases.

RULE 3: <u>All bases must have identical spares inventories</u>.

This is an integral part of the model's assumption that all bases are identical in every respect.

RULE 4: <u>If the purchase quantity is greater than one, but less than NOB, spares must be redistributed from the depot to the bases</u>.

Suppose that more than one spare is purchased. Rule one prohibits us from putting the entire purchase at the depot, and rule three prohibits us from simply putting the entire purchase at the bases, unless the purchase quantity is NOB. To satisfy rule three, we have to put more spares at the bases than we purchased. Therefore, we have to transfer a quantity of spares from the depot to the bases. That quantity is equal to the number of bases (NOB) minus the number of spares that we purchased.

RULE 5: <u>In no case will more than NOB-1 spares be redistributed</u>.

We have not shown that it will never be optimal to redistribute more depot spares; however, we have not observed any cases in which greater redistribution will result in lower levels of EBO.

Applying these rules, it is clear that there can only be NOB+1 permissible purchase alternatives. For a purchase quantity of one, there are only two distributional alternatives--the one spare can be placed at the depot, or the one spare can be placed at the base, with NOB-1 spares being redistributed. And for purchase quantities of more than one, there is just one distributional alternative--all of the purchased spares must go to the bases, and NOB minus the purchase quantity must be redistributed.

COMPUTING EXPECTED BACKORDERS

Most of the time, expected backorders at the base and the depot are computed with an iterative algorithm. We describe that algorithm below as it is used in computing expected backorders at the depot. The same algorithm is used in computing the expected backorders at the bases, the only difference being the names of the variables.

The algorithm that computes DEBO performs the following steps with each iteration:

STEP 1: Compute the probability that the population in the depot pipeline is equal to the inventory on hand. The logarithm of that probability is called DPROB.

STEP 2: Compute the reduction in backorders that will result from the addition of a spare. This quantity is a function of DPROB, and is called DREBO.

STEP 3: Compute the level of depot backorders that will result from the addition of a spare. This variable is called DEBO. The new value of DEBO will equal the existing value of DEBO minus DREBO.

In the discussion below, we will denote the existing value of DEBO as $DEBO_N$, and we will denote the new value as $DEBO_{N+1}$. The same notation will also be used for DREBO and DPROB.

All of these variables can be expressed in forms that lend themselves to iterative computations. Consider DPROB, the logarithm of a Poisson probability. We know that the probability of there being N items in the pipeline is

$$\frac{e^{-\,DMEAN}\ DMEAN^{N}}{N!}$$

Thus $DPROB_N$ is

$$-DMEAN + N\ \ln(DMEAN) - \ln(N!)$$

and $DPROB_{N+1}$ is equal to

$$-DMEAN + (N + 1)\ \ln(DMEAN) - \ln[(N+1)!]$$

or, equivalently,

$$DPROB_N + \ln(DMEAN) - \ln(N+1)$$

DREBO can also be computed iteratively. To see how it can be computed iteratively, consider a situation in which there are no depot spares. A spare at this point will reduce the number of depot backorders to zero when it otherwise would have been one, to one when it otherwise would have been two,

to two when it otherwise would have been three, etc. In short, it reduces DEBO from

$$\sum_{i=o}^{\infty} p_i \, (i)$$

to

$$\sum_{i=1}^{\infty} p_i \, (i - 1)$$

for a net reduction of

$$\text{DREBO}_1 = \sum_{i=1}^{\infty} p_i = 1 - p_o$$

where P_i represents the probability that i units are in the depot pipeline.

Adding a second spare will further reduce DEBO. When the pipeline contains two units, there will now be no backorders; when it contains three units, there will now be only one backorder, etc. The net reduction in DEBO resulting from the second spare can thus be expressed as

$$\text{DREBO}_2 = \sum_{i=2}^{\infty} p_i = 1 - p_o - p_1 = \text{DREBO}_1 - p_1$$

Similarly, the net reduction in DEBO resulting from the N + 1st spare can be expressed as

$$\text{DREBO}_{N+1} = 1 - p_o - p_1 - p_2 - p_3 \cdots p_N = \text{DREBO}_N - p_N$$

Since p_N is simply the antilogarithm of DPROBN, DREBO_{N+1} can be easily computed.

Using the DRAMA variable names, the iterative computation of DEBO can be summarized as follows:

$$\text{DPROB}_N = \text{DPROB}_{N-1} + \ln(\text{DMEAN}) - \ln(N)$$

$$DREBO_{N+1} = DREBO_N - e^{DPROB_N}$$

$$DEBO_{N+1} = DEBO_N - DREBO_{N+1}$$

The appropriate initial values of these variables are as follows:

$$DPROB_o = - DMEAN$$

$$DREBO_o = 1.0$$

$$DEBO_o = DMEAN$$

For the bases a similar computation of EBO is carried out in subroutine POISSY. POISSY's computations can be summarized as follows for any specified number of depot spares:

$$BPROB_N = BPROB_{N-1} + \ln(BMEAN) - \ln (N)$$

$$BREBO_{N+1} = BREBO_N - e^{BPROB_N}$$

$$BEBO_{N+1} = BEBO_N - BREBO_{N+1}$$

The appropriate initial values of these variables are

$$BPROB_o = - BMEAN$$

$$BREBO_o = 1.0$$

$$BEBO_o = BMEAN$$

To save on running time, the model sometimes uses shortcuts instead of this iterative computation. One shortcut, the subroutine CON, computes BPROB, BREBO, and BEBO for an inventory of spares equal to BMEAN minus six standard deviations; the other shortcut, the subroutine LIB, computes BPROB, BREBO, and BEBO for an inventory of spares equal to BMEAN. These routines are regularly called by the subroutine BNEXT. They may also be called by PRELIM, in which case they will compute the depot variables, DPROB, DREBO, and DEBO, as well as the base variables.

The logic in CON is based on the fact that Poisson probabilities are infinitesimal (less than 10^{-7}) when N is less than BMEAN minus six standard deviations (BMEAN minus six standard deviations is called ISIG). Because of

this fact, the first ISIG spares each provide a backorder reduction of approximately one unit, and the value of BEBO, after their purchase, is simply

$$\text{BMEAN} - \text{ISIG} .$$

When the spares inventory is greater than ISIG, but less than BMEAN, CON has to compute $\text{BREBO}_{\text{ISIG}}$ and $\text{BPROB}_{\text{ISIG}}$. The value of $\text{BREBO}_{\text{ISIG}}$ is one, because the Poisson probabilities have all been rounded to zero. The value of $\text{BPROB}_{\text{ISIG}}$ is given by the equation

$$\text{BPROB}_{\text{ISIG}} = -\text{BMEAN} + \text{ISIG} \ln(\text{BMEAN}) - \text{STIRL}$$

where STIRL is equal to the ln(ISIG!) and is computed by the subroutine FACTLN according to Stirling's formula. CON then transfers these values to POISSY, which uses them to iteratively compute BEBO_N. By calling CON prior to POISSY, far fewer iterations are required to compute BEBO_N. This saves a great deal of time, particularly when the mean of the base Poisson distribution is large.

The subroutine LIB is based on the equivalence between the Poisson distribution function and the incomplete gamma function. Because of this equivalence, and because BREBO_{N+1} can be defined as the complement of the Poisson distribution function, BREBO_{N+1} is equal to the complement of the incomplete gamma function evaluated at N. According to a derivation published by Knuth,[4] the complement of the incomplete gamma function is approximately

$$1/2 - \left[\frac{y-2/3}{\sqrt{2\Pi}}\right] x^{-1/2} - \frac{1}{\sqrt{2\Pi}}\left[\frac{23}{270} - \frac{Y}{12} - \frac{Y^3}{6}\right] x^{-3/2}$$

where x is equal to BMEAN, and

where y is equal to BMEAN minus the

inventory level.

[4]Knuth, Donald E., Fundamental Algorithms, Volume I, p.116, Addison-Wesley Inc., Reading, Mass., 1973.

Although this equation can approximate BREBO for various inventory levels, the approximations are most accurate when the inventory level is equal to the truncated value of BMEAN. This is the inventory level that is used in LIB. In the model, this inventory level is called IMEAN.

LIB computes $BPROB_{IMEAN}$ just as CON computes $BPROB_{ISIG}$, by calling FACTLN and applying the appropriate equation. It then computes $BEBO_{IMEAN}$ with the equation below:

$$BEBO_{IMEAN} = (BMEAN-IMEAN)\ (1-BREBO_{IMEAN+1}) + BMEAN \cdot e^{BPROB_{IMEAN}} .$$

When the stockage level at each base exceeds IMEAN, DRAMA calls POISSY to compute the backorder reduction of the remaining stock.

III. USING THE MODEL

INPUTS

The DRAMA main program requires system-level data such as the number of bases and the number of end items at each base, and some component-level data, such as cost and the means of the depot and base Poisson distributions, DMEAN and BMEANØ. Since these last two data elements are not usually available in weapon system data bases, it may be necessary to compute them from elements that are available. These computations are carried out by the preprocessing program called PROLOG. In the sections below, we discuss PROLOG's outputs and the manner in which PROLOG computes DMEAN and BMEANØ.

PROLOG

The standard PROLOG output consists of scenario data describing the number of bases and end items per base, followed by a data line for each component type. A brief description of the necessary data, along with the corresponding FORTRAN variable name, is given below. In addition, a sample PROLOG output appears in Table III-1.

Scenario Data: NOB, SPB, NCOMP

NOB	-	Number of bases supported by the depot.
SPB	-	Number of end items per base.
NCOMP	-	Number of components.

Component Data: I, COST(I), IPS(I), DMEAN(I), BMEANØ(I)

I	-	Counter used to indicate type-I component.
ICOST(I)	-	Type-I unit cost.
IPS(I)	-	Installations per system; number of type-I components installed on each end item.

DMEAN(I) - The expected number of backorders at the depot given zero spares of type I.

BMEANØ(I) - The expected backorders at a base given zero spares of type I.

TABLE III-1. SAMPLE PROLOG OUTPUT (DRAMA INPUT)

17	121..0 70				Scenario Data: NOB, SPB, NCOMP
1	1607.70	1	2.18	8.12	
2	1435	6	9.59	105.53	
3	227	4	326.97	4591.79	
4	1514	1	.96	3.74	Component Data: I, ICOST(I), IPS(I), BMEANØ(I), DMEAN(I)
5	1514	1	.96	3.74	
6	2230	4	2.00	7.78	
7	2230	4	2.00	7.78	
8	318	2	1.77	6.87	

Scenario Data: SYSRAT, NSUBS

SYSRAT - System utilization rate for the particular scenario being modeled.

NSUBS - The number of subsystems in the data base.

Component Data: RATE, DRCT, BRCT, PLT, OST

RATE(I) - Component removal rate per application.

ICND(I) - The percentages of removed units that get condemned.

IRTS(I,4) - Repairable-this-station rates.

DRCT - Depot repair time.

BRCT - Base repair time.

DSRCT - In four-echelon systems, the repair time at the echelon just above the base echelon.

GSRCT - In four-echelon systems, the repair time at the echelon just below the depot echelon.

PLT - Procurement lead time; the time required to replace a condemned unit.

DOST - Order-and-ship time from the depot to the bases.

GSOST - In four-echelon systems, the order-and-ship time from the echelon just below the depot to the bases.

ISS(I) - The subystem in which unit I is installed.

K(ISS) - The ratio of expected subsystem removal rate to the input subsystem removal rate.

An input data base for PROLOG is illustrated below.

TABLE III-2. SAMPLE PROLOG INPUT

17	121	2.0	70	23.2					Scenario data: NOB, SPB, NSUBS, NCOMP, SYSRAT
1.0									Subsystem data: K(ISS)
1.0									
1607.00	1	10	20	30	40	0	1	7.5	Component data: ICOST(I), IPS(I), IRTS(I,1), IRTS(I,2), IRTS(I,3), IRTS(I,4), CND(I), ISS(I), RATE(I)
1435.00	6	5	10	20	40	25	1	22.3	
.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	
.	.	.	.	.	.	.	.	.	

PROLOG computes the rate of removal of a component at an <u>individual base</u> as follows:

$$BRATE(I) = \{K(ISS(I)\cdot RATE(I)\ \}\ \ \{SYSRAT\}\ \ \{IPS(I)\cdot SPB\}$$

This quantity is used to compute the quantities BMEANØ and DMEAN.

A component's depot pipeline (DMEAN(I)) consists of its depot repair pipeline, its condemnation pipeline and, in cases where four echelons are modeled, its "general support" pipeline. Since a pipeline has a population equal to the rate at which parts enter it times the length of time they remain within it, these three pipelines are respectively equal to

$$\text{DPIPE(I)} = \text{IRTS(I,4)} \cdot \text{NOB} \cdot \text{BRATE(I)} \cdot \text{DRCT},$$

$$\text{CPIPE(I)} = \text{ICND(I)} \cdot \text{NOB} \cdot \text{BRATE(I)} \cdot \text{PLT},$$

and $$\text{GPIPE(I)} = \text{IRTS(I,3)} \cdot \text{NOB} \cdot \text{BRATE(I)} \cdot \text{GRCT}.$$

The depot pipeline equals the sum of these three pipelines, namely:

$$\text{DMEAN(I)} = \text{DPIPE(I)} + \text{CPIPE(I)} + \text{GPIPE(I)}.$$

A component's resupply pipeline at an individual base, BMEANØ, consists of a base's share of the depot pipeline plus the base repair, direct-support repair, and order-and-shipping pipelines. These pipelines are respectively

$$\text{BRPIPE} = \text{IRTS(I,1)} \cdot \text{BRATE(I)} \cdot \text{BRCT}$$

$$\text{DRPIPE} = \text{IRTS(I,2)} \cdot \text{BRATE(I)} \cdot \text{DSRCT}$$

$$\text{GOPIPE} = \text{IRTS(I,3)} \cdot \text{BRATE(I)} \cdot \text{GOST}$$

and $$\text{DOPIPE} = [\text{IRTS(I,4)} + \text{ICND(I)}] \cdot \text{BRATE(I)} \cdot \text{DOST}.$$

The resupply pipeline at an individual base is given by

$$\text{BMEAN}\emptyset\text{(I)} = \frac{\text{DPIPE(I)}}{\text{NOB}} + \text{BRPIPE(I)} + \text{DRPIPE(I)} + \text{GOPIPE(IP)} + \text{DOPIPE(I)}.$$

STANDARD DRAMA OUTPUT

Every DRAMA run will output a 27-line cost-availability table. A sample output appears in Table III-3.

OPTIONAL DRAMA OUTPUT

DRAMA will also output special files at the user's request. (The user must attach these extra files to the DRAMA program in order to output data to them. The appendix explains how this can be done on the Honeywell 635).

TABLE III-3. SAMPLE AVAILABILITY-COST RELATIONSHIP

Line No.	Availability	Total Spares Cost	Line No.	Availability	Total Spares Cost
1	5%	386061696	15	75%	584437472
2	10%	387825984	16	80%	614785536
3	15%	389122432	17	85%	655098880
4	20%	393912320	18	90%	707731904
5	25%	396275840	19	91%	719241698
6	30%	399317952	20	92%	731577216
7	35%	402902944	21	93%	744964448
8	40%	408072640	22	94%	759649216
9	45%	417048192	23	95%	776184256
10	50%	441347328	24	96%	795014144
11	55%	471371392	25	97%	817839264
12	60%	500581728	26	98%	847312960
13	65%	528524608	27	99%	892400672
14	70%	556713632			

DRAMA can output four special files. One is the Base Spares File, which contains the quantity of each LRU in the base inventory for each of the 27 availability levels. The second is the Depot Spares File, which contains the quantity of each LRU in the depot inventory for each of the 27 availability levels. The third is the BEBO File, which contains the expected number of missing units for each LRU for each of the 27 availability levels, and the last is the DEBO File, which contains the expected number of outstanding orders at the depot for each LRU.

To obtain these files, the user must specify an output option in the OPTION section of the main program. If he specifies no option, he gets only the standard output. If he sets the variable IOUT equal to one, he will get

only the Base Spares File in addition to the standard output. If he sets IOUT equal to two, he'll get both the Base Spares File and the Depot Spares File. If he sets IOUT equal to three, he'll get all of the extra files.

EPILOG

EPILOG processes DRAMA'S output files and generates detailed tables of LRU costs and backorders. To run EPILOG, the user must input the DRAMA output files and the DRAMA input file (The method for inputting these files is described in the next chapter.) In addition, he must specify his EPILOG output options.

The variable IOUT appears again in EPILOG and, again, its purpose is to dictate what will be output. By setting it equal to one, the user will obtain a listing of the Base Spares File; by setting it equal to two, the user will obtain a listing of the Depot Spares File as well; by setting it equal to three, he'll obtain a comprehensive output table of quantities of component spares at the base and the depot, the cost of the component spares at the base and the depot, the unsatisfied component demand (expected back orders or EBOs) at the base and the depot, and the component availability at the base. A sample EPILOG output for IOUT = 3 is shown in Table III-5. Notice that this output is for a special case where there were initially no depot backorders of any component.

The first column of this table is the LRU reference number; the second is the quantity of spares at the depot; the third is the quantity of spares at each base; the fourth is the total spares in the system; the fifth is the cost of the depot spares; the sixth is the cost of the base spares; the seventh is the total cost; the eighth is the LRU's DEBO; the ninth is the LRU's BEBO and the tenth is the expected LRU availability. In addition to the table, EPILOG

will output system-level summaries of base and depot inventories and costs. The summary for the run that produced Table III-5 appears in Table III-4, below.

TABLE III-4. SUMMARY STATISTICS AT THE SYSTEM LEVEL

Total Spares Purchased	894
Total Spares at the Depot	0
Total Spares at Each Base	298
Total Cost of All Initial Spares	5773752
Total Cost of Spares Bought for the Depot	0
Total Cost of Spares Bought for the Bases	5773752
Base Expected Back Orders	0.01489
Depot Expected Back Orders	0
Full Systems Availability	99.5%

Normally, EPILOG will output a table for each cost-availability pair in DRAMA's standard output. Should the user wish to obtain an output table for just one availability level he can do so by setting NAVAIL equal to the line number (from DRAMA's standard output) of the availability that he desires to see. The line-numbers of each availability are shown in Table III-3. Failure to assign a value to NAVAIL will result in the output of tables for all of the 27 availability levels.

TABLE III-4. SAMPLE PROLOGUE OUTPUT FOR "IOUT" EQUALS 3

********************************SUMMARY OF MODULE PURCHASES******************************

****************FOR A 215 MODULE SYSTEM WITH A DIAGNOSTIC ACCURACY OF 100%*************

MOD	DEPO	BASE	TOT	$ DEPOT $	$ BASES $	$$ TOTAL $$	EBO-DEPOT	EBO-BASES	MOD AVAIL
1	0	2	6	0.	11112.	11112.	0.	0.00007366	0.99997544
2	0	1	3	0.	38958.	38958.	0.	0.00001129	0.99999623
3	0	1	3	0.	4500.	4500.	0.	0.00000040	0.99999987
4	0	2	6	0.	26268.	26268.	0.	0.00001071	0.99999644
5	0	1	3	0.	415683.	415683.	0.	0.00021364	0.99992879
6	0	2	6	0.	30828.	30828.	0.	0.00000421	0.99999860
7	0	1	3	0.	154473.	154473.	0.	0.00161703	0.99946110
8	0	2	6	0.	52836.	52836.	0.	0.00002683	0.99999106
9	0	1	3	0.	1041.	1041.	0.	0.00000835	0.99999722
10	0	1	3	0.	2727.	2727.	0.	0.00000190	0.99999937
11	0	2	6	0.	138294.	138294.	0.	0.00017398	0.99994201
12	0	1	3	0.	600000.	600000.	0.	0.00034082	0.99988640
13	0	2	6	0.	5238.	5238.	0.	0.00000076	0.99999975
14	0	2	6	0.	6000.	6000.	0.	0.00000032	0.99999989
15	0	1	3	0.	3000.	3000.	0.	0.00000002	0.99999999
16	0	2	6	0.	6000.	6000.	0.	0.00000028	0.99999991
17	0	2	6	0.	6000.	6000.	0.	0.00001541	0.99999486
18	0	1	3	0.	3000.	3000.	0.	0.00000055	0.99999982
19	0	1	3	0.	3000.	3000.	0.	0.00004696	0.99998434
20	0	2	6	0.	6000.	6000.	0.	0.00000111	0.99999962
21	0	1	3	0.	3000.	3000.	0.	0.00005108	0.99998297
22	0	1	3	0.	3000.	3000.	0.	0.00000093	0.99999969
23	0	2	6	0.	858.	858.	0.	0.00000239	0.99999920
24	0	1	3	0.	3000.	3000.	0.	0.00000022	0.99999993
25	0	1	3	0.	3000.	3000.	0.	0.00001932	0.99999356
26	0	1	3	0.	3000.	3000.	0.	0.00000001	1.00000000
27	0	1	3	0.	3000.	3000.	0.	0.00000006	0.99999998
28	0	1	3	0.	3000.	3000.	0.	0.00000104	0.99999965
29	0	1	3	0.	3000.	3000.	0.	0.00006141	0.99997953
30	0	1	3	0.	3000.	3000.	0.	0.00000110	0.99999964
31	0	1	3	0.	3000.	3000.	0.	0.00002016	0.99999328
32	0	1	3	0.	3000.	3000.	0.	0.00001908	0.99999363
33	0	1	3	0.	3000.	3000.	0.	0.00000010	0.99999996
34	0	1	3	0.	3000.	3000.	0.	0.00000020	0.99999993
35	0	1	3	0.	3000.	3000.	0.	0.00000001	0.99999999
36	0	1	3	0.	3000.	3000.	0.	0.00003758	0.99998748
37	0	1	3	0.	3000.	3000.	0.	0.00001424	0.99999526
38	0	1	3	0.	3000.	3000.	0.	0.00000185	0.99999938
39	0	1	3	0.	3000.	3000.	0.	0.00000037	0.99999987
40	0	1	3	0.	3000.	3000.	0.	0.00000051	0.99999983
41	0	1	3	0.	3000.	3000.	0.	0.00000000	1.00000000
42	0	1	3	0.	3000.	3000.	0.	0.00000039	0.99999987
43	0	1	3	0.	3000.	3000.	0.	0.00005265	0.99998245
44	0	1	3	0.	3000.	3000.	0.	0.00000051	0.99999983
45	0	1	3	0.	3000.	3000.	0.	0.00000006	0.99999998
46	0	0	0	0.	0.	0.	0.	0.00005032	0.99998321
47	0	1	3	0.	3000.	3000.	0.	0.00000086	0.99999971
48	0	1	3	0.	3000.	3000.	0.	0.00000130	0.99999957
49	0	2	6	0.	6000.	6000.	0.	0.00000105	0.99999965

APPENDIX A

GLOSSARY OF DRAMA VARIABLES

AM	In LIB or CON, the mean of Poisson distribution of backorders at either the depot or the bases.
AVCOMP	The availability of a particular component in EPILOG.
AVLOG(I)	The logarithm of a component's availability.
AVTEST	The next availability milestone at which availability and cost data will be output. Normally, it is initialized at five per cent.
AVTOT	A particular availability level in EPILOG.
BBOLIB(N)	In MARGIN, the value that BEBO will assume if purchase alternative N is selected.
BBOTOT	BEBO summed across all components at a particular availability level. Used only in EPILOG.
BCOST	The total cost of base spares at a particular availability level. Used only in EPILOG.
BEBO(I)	The expected number of backorders of component (I) at each base.
BMLOG	In BNEXT, the logarithm of the variable BMEANN, which denotes the mean value of the base Poisson distribution for purchase alternative N.
BMEANØ(I)	The expected number of units of component (I) in resupply to each base. Initially, when a component has no spares, it is also equal to BEBO(I). It is called a "mean" because it is the mean value of the Poisson distribution of backorders at each base when the depot has zero spares.
BMEANN	In MARGIN, the mean of the Poisson distribution of backorders at each base for purchase alternative N.
BPROB	In BNEXT, this variable denotes the Poisson probability that exactly K spares will be backordered.
BRATE	The expected number of component removals per month at each base. Used only in PROLOG.

BRCT	Base repair-cycle time. The time that elapses from the removal of a base-repairable unit to its return to serviceable condition. Used only in PROLOG.
BREBO	In BNEXT, the reduction in backorders that will be experienced if an additional spare is added to each base.
BRPIPE	The base-repair pipeline. Used only in PROLOG.
C1	A constant used in the subroutine LIB in the computation of either depot or base EBO. It equals two-thirds.
C2	Another constant used in the subroutine LIB in the computation of depot or base EBO. It equals 23/270.
C3	Another constant used in subroutine LIB in the computation of depot or base EBO. It equals one-sixth.
CPIPE	The condemnation pipeline. Used only in PROLOG.
CRANK(I)	The benefit-to-cost ratio, or ranking value, of a component.
DBOLIB(N)	In MARGIN, the value that DEBO will assume if purchase alternative N is best.
DBOTOT	DEBO summed across all components at a particular availability level. Used only in EPILOG.
DCOST	The total cost of depot spares at a particular availability level. Used only in EPILOG.
DEBO(I)	The expected number of backorders of a component at the depot.
DELTA	In AVAIL, DELTA equals the difference between the availability logarithm of the purchased component <u>after</u> its purchase and the availability logarithm of the component <u>before</u> its purchase.
DMEAN(I)	The expected number of units of a component in shipment to the depot, depot repair, or procurement. This variable is the mean of the Poisson distribution of backorders at the depot.
DMLOG	The log of the depot pipeline, DMEAN. It is used in the computation of DEBO.
DOPIPE	The depot order-and-shipping pipeline for each base. Used only in PROLOG.
DOST	The depot order-and-shipping time. Used only in PROLOG.
DPIPE	The depot repair pipeline. Used only in PROLOG.

DPQ	In FACTLN, the double-precision equivalent of the variable ISIG.
DPROB(I)	The logarithm of a Poisson probability. Specifically, the log of the probability that the population of the depot pipeline is equal to JSPARE(I).
DRBLIB(N)	In MARGIN, the reduction in backorders that will be provided by an additional depot spare if purchase alternative N is best.
DRCT	Depot repair-cycle time. The time that elapses from the removal of a depot-repairable unit to its return to serviceable condition. Used only in PROLOG.
DREBO(I)	The reduction in expected backorders that will accompany the addition of a depot spare. This variable strictly decreases as the depot inventory increases.
DSPIPE	The direct-support repair pipeline. Used only in PROLOG.
DSRCT	The direct-support repair-cycle time. The time that elapses from the removal of a Direct-Support repairable unit to its return to serviceable condition.
E	In LIB, CON, POISSY or PRELIM, the EBO that will persist at a site after it has obtained a pipeline's worth of spares.
GOPIPE	The general-support order-and-shipping pipeline. Used only in PROLOG.
GSOST	The general-support order-and-shipping time. Used only in PROLOG.
GSPIPE	The general-support pipeline. Used only in PROLOG.
GSRCT	General-support repair-cycle time. The time that elapses from the removal of a general-support repairable unit to its return to serviceable condition. Used only in PROLOG.
I	Indicates a particular component. It takes on values between (and including) component and NCOMP.
ICND	The percentage of parts condemned. Used only in PROLOG.
ICOST(I)	The cost of a spare unit of a component.
IDONE	Indicates whether all components have been ranked in COMPOP. It is initially equal to zero; it is set equal to one once the ranking process is completed.
IFIN	In POISSY, the quantity of base spares accounted for when the iterative computation of EBO ends.
IMEAN	In LIB or CON, the truncated value of AMEAN.

IOUT	When it equals zero, DRAMA provides only standard output. When it equals one, the base spares file is output. When it equals two, the depot spares file is output as well. When it equals three, depot and base EBO files will be output too.
IPOS(NCOMP)	Indicates which component is in ranking position one, ranking position two, etc.
IPRE	Dictates whether to call subroutine PRELIM. When it equals zero, PRELIM is bypassed; when it equals one, PRELIM buys conservatively. When it equals two, PRELIM buys liberally.
IPS(I)	The number of parts installed per end item or system.
IRTS1	The percentage of parts repaired at the base echelon. Used only in PROLOG.
IRTS2	The percentage of parts repaired at the direct-support echelon. Used only in PROLOG.
IRTS3	The percentage of parts repaired at the general-support echelon. Used only in PROLOG.
IRTS4	The percentage of parts repaired at the depot echelon. Used only in PROLOG.
ISIG	The truncation of the following quantity: The Base or Depot pipeline minus six standard deviations.
ISP	In CON, the quantity of spares at each base; a surrogate for KSP.
ISTAR	In POISSY, the quantity of base spares accounted for when the iterative computation of BEBO begins.
ISTRT	In CON, the quantity of base (or depot) spares for which an EBO is computed. It equals either the total inventory of base (or depot) spares or the base (or depot) pipeline minus six standard deviations, whichever is less.
IT	The line in the DRAMA availability-cost table that is being processed in EPILOG.
ITERS	Counts the number of marginal analysis iterations in a run.
JJ	In DNEXT, it indicates how many spares will be redistributed in a given purchase alternative.
JK	In DNEXT, it is the indicator of the purchase alternative.
JSPARE(I)	The number of spares of a component at the depot.
K1	A constant used in subroutine FACTLN. It equals one-half of the square root of TWOPI.

K2	A constant used in FACTLN. It equals one-twelfth.
KSP	In BOUT, the number of spares that each base will have after a given purchase alternative is executed.
KSPARE(I)	The number of spares of a component at each base.
KSTRT	The number of base spares that BNEXT has accounted for in the process of computing BBOLIB.
K(NSUBS)	The subsystem "K-factor." The ratio of subsystem removals in the DRAMA data base to the subsystem removals in the raw data base. Used only in PROLOG.
LCOST	The cost of a particular component's spares in EPILOG.
LLL	In PRELIM, the truncated value of PM.
LNFACT(30)	In FACTLN, an array of the logarithms of the factorials of 1 through 30.
M	In BOUT, the indicator of the purchase alternative being considered.
MAXRED	The maximum number of parts that may be redistributed from the depot to the bases in a given purchase.
MODE	When it equals one, MARGIN computes a backorder-reduction to cost-ratio, instead of a marginal availability-to-cost ratio, for each purchase alternative. It is set equal to one only when a component's marginal availability is undefined, e.g., when it has more backorders at each base than applications.
N	In MARGIN, it indicates the purchase alternative. It takes on values from 1 to NOB+1.
NAVAIL	The line in the DRAMA availability-cost table that is to be output by EPILOG. When it is equal to zero, all lines are to be output.
NBEST(I)	The purchase alternative which will yield the most availability for its cost.
NCOMP	The number of different types of line-replaceable units (LRUs) in an end-item.
NEWPOS	In COMPOP the new ranking position of the component that was just purchased.
NOB	The number of bases.
NTABLE	The number of cost-availability pairs that have to be input to EPILOG.

P	In LIB, the Poisson probability that the quantity of parts in the depot (or base) pipeline is exactly equal to the mean of the pipeline.
PLT	Procurement lead time. The time that elapses from the condemnation of a component to the receipt of a replacement at the depot.
PO	In PRELIM, LIB, or CON, the temporary value of BMLOG or DMLOG.
Q	In LIB, an intermediate value in the computation of depot or base EBO.
QRANK(N)	In SIZOP, the marginal availability (or marginal EBO reduction)-to-cost ratio for purchase alternative N.
R	In LIB, or CON, the reduction in BEBO that will accompany the addition of one spare to each base.
RATE	The frequency of component removals. The number of removals per component application per hour of use. Used only in PROLOG.
SPB	The number of systems (or end items) per base.
STIRL	In FACTLN, LIB, and CON, the logarithm of the factorial of KSP, where KSP is the number of spares accounted for at each base.
SYSLOG	The logarithm of the system's availability.
SYSRAT	The system utilization rate. Expressed in usage hours per month. Used only in PROLOG.
SYSTAV	The system availability.
TAIL	In LIB, the area under the Poisson probability density function between IMEAN and infinity.
TCOST	The total cost of spares at a particular availability level. Used only in EPILOG.
TEMP(NCOMP)	In COMPOP, the component that is temporarily ranked number one, number two, etc.
TOTCST	The total cost of spares of all component types.
TWOPI	Another constant in the subroutine LIB. It equals two times Pi.
Y	In LIB, an intermediate value in the computation of base or depot EBO.

APPENDIX B

PROGRAM LISTINGS

PROLOG

```
list

      REAL K(30)
      CALL FMEDIA(9,0)
      CALL FMEDIA(10,0)
C
C*****SCENARIO DATA*****
C
      READ(9,109) NOB,SPB,NSUBS,NCOMP,SYSRAT
C
C*****SUBSYSTEM DATA*****
C
      DO 1000 ISS = 1,NSUBS
 1000 READ(9,109) K(ISS)
  109 FORMAT(V)
C
C*****ASSIGN VALUES TO THE REPAIR AND SHIPPING TIMES.*****
C
C     ***NORMALLY, THESE TIMES ARE EXPRESSED IN MONTHS***
C
C          **NOTE:TO MODEL A PERFECT TWO-ECHELON SYSTEM, SET
C            DSRCT,GSRCT,DSOST,AND GSOST TO ZERO.
C
      BRCT = .1
      DSRCT = .5
      GSRCT = 1.0
      DRCT = 5.0
      PLT = 18.0
      GSOST = .5
      DOST = 2.0
C
C*****WRITE SCENARIO INPUT TO DRAMA INTO FILE CODE 10.*****
C
     WRITE(10,109) NOB,SPB,NCOMP
C
C*****NOW INPUT THE COMPONENT DATA.  PROCESS IT. THEN OUTPUT
C    IPS,ICOST,BMEANO,AND DMEAN TO FILE CODE 10.
C
      DO 2000 I = 1,NCOMP
      READ(9,109) ICOST,IPS,IRTS1,IRTS2,IRTS3,IRTS4,ICND,ISS,RATE
C
C     *****COMPUTE BRATE FOR THIS COMPONENT.*****
      BRATE = K(ISS) * RATE * SYSRAT * FLOAT(IPS) * SPB
C
C     *****COMPUTE DMEAN FOR THIS COMPONENT.*****
      DPIPE = FLOAT(NOB) * FLOAT(IRTS4/100) * BRATE * DRCT
      CPIPE = FLOAT(NOB) * FLOAT(ICND/100) * BRATE * PLT
      GSPIPE = FLOAT(NOB) * FLOAT(IRTS3/100) * BRATE * GSRCT
      DMEAN = DPIPE + CPIPE + GSPIPE
C
C     *****COMPUTE BMEANO FOR THIS COMPONENT.*****
      BRPIPE = FLOAT(IRTS1/100) * BRATE * BRCT
      DSPIPE = FLOAT(IRTS2/100) * BRATE * DSRCT
      GOPIPE = FLOAT(IRTS3/100) * BRATE * GSOST
```

```
           DOPIPE = FLOAT( (IRTS4+ICND)/100 ) * BRATE * DOST
           BMEANO = DMEAN/FLOAT(NOB) + BRPIPE + DSPIPE + GOPIPE +
     &                DOPIPE
C
C*****OUTPUT COMPONENT DATA.*****
 2000      WRITE(10,109) IPS,ICOST,BMEANO,DMEAN
           STOP
           END
```

*

DRAMA

LIST

```
1000       COMMON/AVA/AVLOG(70),CRANK(70),IPS(70)
1010       COMMON/BASE/BEBO
1020C       COMMON/EBON/BEBON(70),DEBON(70)
1030       COMMON/CONST/C1,C2,C3,TWOPI
1040       COMMON/DEPOT/DEBO,DMLOG,DPROB,DREBO
1050       COMMON/KONST/K1,K2
1060       COMMON/NEXT/BBOLIB,DBOLIB,PROLIB,DRBLIB
1070       COMMON/MEAN/BMEANO(70),DMEAN(70)
1080       COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
1090       COMMON/PRICE/ICOST(70),NBEST(70)
1100       COMMON/SPARE/JSPARE(70),KSPARE(70)
1110       COMMON/SYST/SYSLOG,SYSTAV,TOTCST
1120       COMMON/OPTS/IPRE
1130        REAL BEBO(70)
1140        REAL DEBO(70),DMLOG(70),DPROB(70),DREBO(70)
1150        DOUBLE PRECISION C1,C2,C3,TWOPI
1160        DOUBLE PRECISION K1,K2
1170        DOUBLE PRECISION BBOLIB(18),DBOLIB(18),PROLIB(18),DRBLIB(18)
1180C
1190C*****THIS SECTION OF DRAMA ATTACHES THE NECESSARY DATA FILES.
1200C      THE FIRST FILE ATTACHED IS THE INPUT FILE.  THE LAST FILES
1210C      ATTACHED ARE OPTIONAL OUTPUT FILES.
1220        CALL ATTACH(9,"0S29/N232D/LEMON;",3,0,ISTAT, )
1230C       CALL FMEDIA(10,0)
1240C       CALL FMEDIA(11,0)
1250C       CALL FMEDIA(12,0)
1260C       CALL FMEDIA(13,0)
1270C
1280C*****THIS SECTION OF DRAMA ASSIGNS INITIAL VALUES TO ALL KEY
1290C      KEY VARIABLES.  USERS MUST ASSIGN VALUES HERE IF THEY
1300C      WISH TO EXECUTE OPTIONS
1310C
1320C      *****USER DEFINED CONTROL VARIABLES*****
1330        IPRE = 0
1340        IOUT = 0
1350C      *****NOTE: IF THE USER SETS IOUT TO 3 OR 4 HE MUST DO TWO
1360C           THINGS IN ORDER TO OUTPUT EXACT VALUES OF BEBO AND DEBO.
1370C           ONE, HE MUST DEFINE "BEBON" AND "DEBON" IN THE COMMON
1380C           SECTIONS OF DRAMA (SEE LINE 1020), MARGIN, AND UPDATE.
1390C           TWO, HE MUST RECOMPUTE BEBO IN SUBROUTINE UPDATE INSTEAD
1400C           OF SUBROUTINE MARGIN.  SEE COMMENTS BELOW.
1410        AVLIM = .999
1420        COLIM = 99999999.
1430C
1440C      *****INTERNALLY DEFINED CONTROL VARIABLES*****
1450        ITERS = 0
1460        IDONE = 0
1470        AVTEST = AVTEST + .05
1480C
1490C      *****SYSTEM LEVEL SCENARIO VARIABLES*****
1500        READ(9,109) NOB,SPB,NCOMP
1510C
```

```
C      *****COMPONENT'LEVEL VARIABLES*****
        DO 900 I = 1,NCOMP
  900   READ(9,109) IPS(I),ICOST(I),BMEANO(I),DMEAN(I)
  109   FORMAT(V)
C            ***NOTE: IN THIS FORMAT, ONE SPACE SEPARATES EACH DATA
C               ELEMENT FROM ITS SUCCEEDING ELEMENT.
C
C      *****SUBROUTINE CONSTANTS*****
        C1 = 2.0D0/3.0D0
        C2 = 23.0D0/270.0D0
        C3 = .25D0 * C1
        TWOPI = 2.0D0 * 3.14159265358973 2D0
        K1 = .5D0 * DLOG(TWOPI)
        K2 = .5D0 * C3
C
C*****WRITE OUTPUT HEADER
        WRITE(6,996)
 996  FORMAT(' ','AVAILABILITY',16X,'COST')
C
C
C*****THIS SECTION OF DRAMA INITIALIZES THE VARIABLES INVOLVED IN
C      THE COMPUTATION OF BEBO AND DEBO.  THE DO LOOP BELOW ALSO
C      INITIALIZES COMPONENT AVAILABILITIES AND COMPUTES THE INI-
C      TIAL COMPONENT RANKING VALUES.
C
        DO 1000 I = 1,NCOMP
C          ***NOTE: THE VARIABLE "MODE" TELLS THE SUBROUTINE MARGIN
C             WHETHER A COMPONENT HAS A POSITIVE AVAILABILITY.  IF A
C             COMPONENT HAS A NON-POSITIVE AVAILABILITY, THEN
C             MODE STAYS EQUAL TO 2, AND AN AUTOMATIC PRELIMINARY
C             PURCHASE WILL BE MADE IN ORDER TO MAKE AVAILABILITY
C             POSITIVE.
         MODE = 1
C
C      *****INITIALIZE SPARES INVENTORIES*****
        JSPARE(I) = 0
        KSPARE(I) = 0
C
C      *****INITIALIZE THE VARIABLES INVOLVED IN THE EBO COMPUTA-
C           TIONS
        BEBO(I) = BMEANO(I)
        DEBO(I) = DMEAN(I)
        DMLOG(I) = DLOG(DMEAN(I))
        DPROB(I) = -DMEAN(I)
        DREBO(I) = 1.0D0
C          ***NOTE: THE INDEX OF A DO-LOOP CANNOT BE PASSED TO
C           A SUBROUTINE.  THEREFORE, IMAX MUST BE USED AS A SUR-
C            GATE FOR "I".
        IMAX = I
C
C*****MAKE A PRELIMINARY BUY OF THIS COMPONENT IF ONE IS CALLED FC
        IF ( IPRE .GT. 0 .AND. DMEAN(I) .GE. 10.0 ) CALL PRELIM(IMA
C
C*****INITIALIZE THIS COMPONENT'S AVAILABILITY*****
```

```
      CALL AVAIL(IMAX)
C
C*****COMPUTE THIS COMPONENT'S RANKING VALUE*****
C
 1000 CALL MARGIN(IMAX)
C
C
C*****COMPLETE THE FIRST ITERATION OF MARGINAL ANALYSIS.  RANK ALL
C     THE COMPONENTS AND IDENTIFY THE BEST BUY.  WHEN DONE, RESET
C     "IDONE".
      CALL COMPOP(IMAX)
      IDONE = 1
C
C*****PURCHASE THE BEST COMPONENT*****
 2000 CALL UPDATE(IMAX)
      ITERS = ITERS + 1
C     *****TEST:SHOULD AVAILABILITY AND COST DATA BE OUTPUT AT
C          THIS POINT?
      IF ( SYSTAV .LT. AVTEST ) GO TO 5000
      IF ( AVTEST .GE. .989 ) AVTEST = AVTEST + .001
      IF ( AVTEST .GE .89 .AND. AVTEST .LT. .851 ) AVTEST =
     &           AVTEST + .01
      IF ( AVTEST .GE. .05 .AND. AVTEST .LT. .851 ) AVTEST =
     &       AVTEST + .05
C     *****OUTPUT DATA IF THE TEST IS POSITIVE*****
      WRITE(6,1006) SYSTAV,TOTCST
 1006 FORMAT(' ',F12.8,10X,F10.0)
C     *****OPTIONAL OUTPUTS*****
C      IF ( IOUT .GT. 0 ) WRITE(10) KSPARE
C      IF ( IOUT .GT. 1 ) WRITE(11) JSPARE
C      IF ( IOUT .GT. 2 ) WRITE(12) BEBO
C      IF ( IOUT .GT. 2 ) WRITE(13) DEBO
C
C
C*****TEST: HAS A CONSTRAINT BEEN REACHED?*****
 5000 CONTINUE
      IF ( SYSTAV .GE. AVLIM .OR. TOTCST .GE. COLIM )
     &       GO TO 6000
C*****IF NEITHER CONSTRAINT HAS BEEN REACHED REPEAT MARG. ANALYSIS.
      CALL MARGIN(IMAX)
      CALL COMPOP(IMAX)
      GO TO 2000
C*****IF EITHER CONSTRAINT HAS BEEN REACHED, STOP.*****
 6000 CONTINUE
      STOP
      END
C-----------------------------------------------------------------
      SUBROUTINE MARGIN(I)
      COMMON/AVA/AVLOG(70),CRANK(70),IPS(70)
      COMMON/BASE/BEBO
C      COMMON/EBON/BEBON(70),DEBON(70)
      COMMON/DEPOT/DEBO,DMLOG,DPROB,DREBO
      COMMON/NEXT/BBOLIB,DBOLIB,PROLIB,DRBLIB
      COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
```

```
      COMMON/PRICE/ICOST(70),NBEST(70)
       REAL BEBO(70)
       REAL DEBO(70),DMLOG(70),DPROB(70),DREBO(70)
       DOUBLE PRECISION BBOLIB(18),DBOLIB(18),PROLIB(18),DRBLIB(18)
C
C*****TEST: CAN BEBO BE FURTHER REDUCED?  IF IT CANNOT,
C     SKIP BEBO COMPUTATIONS AND SET THE RANKING VALUE TO ZERO.
C
      IF ( BEBO(I) .LE. 0.0 ) GO TO 1510
C
C     *****COMPUTE BBOLIB*****
C
 2010    CALL DNEXT(I)
         CALL BNEXT(I)
C
C*****BEBO OUTCOMES ARE NOW STORED IN THE ARRAY "BBOLIB".  USE
C     THEM TO COMPUTE THE BENEFIT-TO-COST RATIOS OF THE FEAS-
C     IBLE PURCHASES.  THEN FIND THE MOST COST-EFFECTIVE ONE.
C
         CALL SIZOP(I)
C
C*****THE MOST COST EFFECTIVE PURCHASE HAS BEEN DESIGNATED
C     "NBEST".  STORE THE DEBO AND BEBO VALUES ASSOCIATED
C     WITH THIS PURCHASE.
C
       BEBO(I) = BBOLIB( NBEST(I) )
C       BEBON(I) = BBOLIB(NBEST(I))
       DEBO(I) = DBOLIB( NBEST(I) )
C       DEBON(I) = DBOLIB(NBEST(I))
       DPROB(I) = PROLIB( NBEST(I) )
       DREBO(I) = DRBLIB( NBEST(I) )
C
C*****IF THIS COMPONENT ALREADY HAS A DEFINED AVAILABILITY, RETURN.
C     IF IT DOES NOT, BUY SPARES UNTIL ITS AVAILABILITY CAN BE
C     COMPUTED.
C
          IF ( MODE .EQ. 2 ) GO TO 1610
          CALL UPDATE(I)
          GO TO 2010
 1510  CRANK(I) = 0.0
 1610  CONTINUE
       RETURN
         END
C------------------------------------------------------------------
      SUBROUTINE DNEXT(I)
       COMMON/NEXT/BBOLIB,DBOLIB,PROLIB,DRBLIB
       COMMON/DEPOT/DEBO,DMLOG,DPROB,DREBO
       COMMON/SPARE/JSPARE(70),KSPARE(70)
       COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
        REAL DEBO(70),DMLOG(70),DREBO(70),DPROB(70)
        DOUBLE PRECISION BBOLIB(18),DBOLIB(18),DRBLIB(18),PROLIB(18)
C
C*****COMPUTE THE DEBO THAT WILL RESULT IF THE DEPOT GETS A
C     SPARE.  THIS IS PURCHASE ALTERNATIVE NOB+1.
```

```
C
       N = NOB + 1
        DRBLIB(N)  = DREBO(I)  - DEXP(DPROB(I))
        DBOLIB(N)  = DEBO(I)  - DRBLIB(N)
        PROLIB(N)  = DPROB(I)  - DLOG( 1.0D0 + DBLE(JSPARE(I))) +
     &                 DMLOG(I)
C
C*****STORE THE CURRENT VALUES OF DEBO, DPROB, AND DREBO.  THIS IS
C     PURCHASE ALTERNATIVE NOB.
C
        DBOLIB(NOB)  = DEBO(I)
        DRBLIB(NOB)  = DREBO(I)
        PROLIB(NOB)  = DPROB(I)
C
C*****COMPUTE THE VALUES OF DEBO, DPROB, AND DREBO FOR ALL OF THE
C     OTHER PURCHASE ALTERNATIVES.
C
C     *****DETERMINE THE MAXIMUM NUMBER OF SPARES THAT CAN BE RE-
C          DISTRIBUTED.  IF NONE CAN BE REDISTRIBUTED, RETURN.
C
           MAXRED = MIN0( NOB - 1, JSPARE(I) )
           IF ( MAXRED .EQ. 0 ) GO TO 5015
C
C     *****PERFORM COMPUTATIONS FOR EACH PURCHASE ALTERNATIVE.  DO
C          ALTERNATIVE NOB-1 FIRST, THEN DO NOB-2, NOB-3, ETC. .
C
         DO 4415 JJ = 1, MAXRED
           J = NOB - JJ
           JK = J + 1
           DBOLIB(J)  = DBOLIB(JK)  + DRBLIB(JK)
           PROLIB(J)  = PROLIB(JK)  + DLOG(DBLE(JSPARE(I)) -
     &                    DBLE(JJ) + 1.0D0 ) - DMLOG(I)
           DRBLIB(J)  = DRBLIB(JK)  + DEXP(PROLIB(J))
 4415      IF ( DRBLIB(J) .GE. 1.0D0 ) DRBLIB(J) = 1.0D0
 5015 CONTINUE
      RETURN
      END
C---------------------------------------------------------------------
      SUBROUTINE BNEXT(I)
       COMMON/NEXT/BBOLIB,DBOLIB,PROLIB,DRBLIB
       COMMON/MEAN/BMEANO(70),DMEAN(70)
       COMMON/SPARE/JSPARE(70),KSPARE(70)
       COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
        DOUBLE PRECISION BBOLIB(18),DBOLIB(18),PROLIB(18),DRBLIB(18)
        DOUBLE PRECISION BMEANN,BPROB,BREBO,BMLOG
C
C*****COMPUTE THE BASE SPARES INVENTORY FOR THE NEXT PURCHASE
C
       KSP  = KSPARE(I)  + 1
C
C
C*****COMPUTE THE BASE EXPECTED BACKORDER (BBOLIB) VALUE FOR
C     EACH PURCHASE ALTERNATIVE.
C
```

```
      DO 2020 N = NOB - MAXRED, NOB + 1
         IF ( N .EQ. NOB + 1 ) KSP = KSP - 1
C
C     *****RE-INITIALIZE BMEAN FOR EACH PURCHASE OUTCOME.
C
         BMEANN = DBLE(BMEANO(I)) - ( DBLE(DMEAN(I)) -
     &                DBOLIB(N) )/DBLE(NOB)
C
C     *****IF THE BASE HAS NO SPARES, AND IF IT IS ALLOTTED NO
C          SPARES IN THIS OUTCOME, THEN BBOLIB EQUALS BMEAN.
C
         BBOLIB(N) = BMEANN
         IF( KSP .EQ. 0 ) GO TO 2020
C
C     *****IF THE BASE HAS SPARES, BBOLIB WILL BE LESS THAN BMEAN.
C          COMPUTE BBOLIB FROM BMEAN GIVEN KSP BASE SPARES.
C          DO THIS IN THREE STEPS.  FIRST, INITIALIZE KEY VARIABLES.
C          SECOND, USE SHORTCUT CALCULATIONS, IF POSSIBLE.
C          THIRD, USE AN ITERATIVE COMPUTATION, IF NECESSARY.
C
C          *****INITIALIZE
C
               KSTRT = 0
               BMLOG = DLOG(BMEANN)
               BPROB = -BMEANN
               BREBO = 1.0D0
C
C          *****SHORTCUTS*****
C
C              SHORTCUT 1 COMPUTES THE VALUE OF BBOLIB FOR AN IN-
C              VENTORY OF BMEAN.  THEN IT SETS KSTRT EQUAL TO THIS
C              INVENTORY QUANTITY.
C
               IF ( KSP .GE. IFIX(BMEANN) .AND. BMEANN .GE. 10.0)
     &             CALL LIB(BPROB,BREBO,BBOLIB(N),BMLOG,BMEANN,KSTR
C
C             **SHORTCUT 2 COMPUTES THE VALUE OF BBOLIB FOR AN
C               INVENTORY OF KSP, OR BMEAN MINUS SIX STANDARD DE-
C               VIATIONS, WHICHEVER IS LESS.  THEN IT SETS KSTRT
C               EQUAL TO THIS INVENTORY QUANTITY
C
                IF( KSP .LT. IFIX(BMEANN) .AND. BMEANN .GE. 38.0)
     &             CALL CON(BPROB,BBOLIB(N),BMLOG,BMEANN,KSTRT,KSP)
C
C          *****ITERATIVE COMPUTATION OF BBOLIB.
C
C              IF THE INVENTORY IS GREATER THAN KSTRT, ONLY PART OF
C              THE INVENTORY HAS BEEN ACCOUNTED FOR. THUS
C              POISSY MUST BE CALLED TO ACCOUNT FOR THE REMAINDER.
C
                   IF ( KSP .GT. KSTRT ) CALL POISSY( BPROB,BREBO,
     &                 BBOLIB(N),BMLOG,KSTRT,KSP)
C
 2020    CONTINUE
```

```
          RETURN
          END
C------------------------------------------------------------------------
      SUBROUTINE LIB(P,R,E,PO,AM,IMEAN)
       COMMON/CONST/C1,C2,C3,TWOPI
        DOUBLE PRECISION C1,C2,C3,TWOPI
        DOUBLE PRECISION P,R,E,PO,AM,Y,Q,TAIL,STIRL
C
        IMEAN = IFIX(AM)
C
C*****COMPUTE THE LN OF IMEAN FACTORIAL*****
C
      CALL FACTLN(STIRL,IMEAN)
C
C*****COMPUTE THE POISSON DPROBABILITY THAT IMEAN UNITS ARE IN THE
C     PIPELINE.
C
      P = DBLE(IMEAN) * PO - AM - STIRL
C
C*****COMPUTE THE AREA UNDER THE PDF FROM IMEAN TO INFINITY.
C     USE THE ALGORITHM FROM KNUTH, VOLUME 1, P. 116.
C
          Y = AM - DBLE(IMEAN)
          Q = Y - C1 + ( C2 - C3 * Y / 2.0D0 - C3 * Y * Y * Y )/
     &           DBLE(IMEAN)
          TAIL = Q / DSQRT( DBLE(IMEAN) * TWOPI ) + .5D0
C
C*****R IS THE AREA UNDER THE PDF  FROM IMEAN TO INFINITY IN-
C     CLUDING THE PD OF IMEAN.
C
          R = TAIL + DEXP(P)
C
C*****COMPUTE THE EBO AT THIS SITE GIVEN IMEAN SPARES.
C
      E = Y * TAIL + AM * ( R - TAIL )
C
      RETURN
      END
C------------------------------------------------------------------------
      SUBROUTINE CON(P,E,PO,AM,ISTRT,ISP)
        DOUBLE PRECISION P,E,PO,AM,STIRL
C
C*****COMPUTE THE INTEGER THAT IS SIX STD. DEVIATIONS BELOW THE MEAN.
C
      ISIG = IFIX( AM - 6.0 * SQRT(AM) )
C
C     *****SELECT THE SPARES QUANTITY FOR WHICH EBO WILL
C          BE CALCULATED.
C
           ISTRT = MINO ( ISIG, ISP )
C
C*****DETERMINE UNSATISFIED LIBAND.
C
      E = AM - DBLE(ISTRT)
```

```
           IF ( ISP .LT. ISIG ) GO TO 3030
C
C*****IF THE SPARES INVENTORY IS .GE. ISIG, COMPUTE THE POISSON
C          DPROBABILITY THAT THE PIPELINE POPULATION IS EXACTLY EQUAL
C          TO ISIG.
C
C          *****COMPUTE THE LOG OF ISIG FACTORIAL .
C
                  CALL FACTLN(STIRL,ISIG)
C
C          *****COMPUTE THE LOG OF THE POISSON DPROBABILITY
C
                  P = DBLE(ISIG) * PO - AM - STIRL
C
      3030   CONTINUE
             RETURN
             END
C-----------------------------------------------------------------------
             SUBROUTINE FACTLN(STIRL,IP)
C******* THIS SUBROUTINE GENERATES THE LN OF N FACTORIAL ******
C******* IF N IS LESS THAN 31, THE LOG OF ITS FACTORIAL IS
C              STORED IN THE TABLE BELOW.
C
             COMMON/KONST/K1,K2
             DOUBLE PRECISION DPO,STIRL,K1,K2
             DOUBLE PRECISION LNFACT(30)
             DATA LNFACT/
         &                     0.0D0,
         & .69314718055994531D0,
         & .179175946922805500D1,
         & .317805383034794562D1,
         & .478749174278204599D1,
         & .657925121201010099D1,
         & .852516136106541430D1,
         & .106046029027452502D2,
         & .128018274800814696D2,
         & .151044125730755153D2,
         & .175023078458738858D2,
         & .199872144956618862D2,
         & .225521638531234229D2,
         & .251912211827386815D2,
         & .278992713838408916D2,
         & .306718601060806728D2,
         & .335050734501368889D2,
         & .363954452080330536D2,
         & .393398841871994940D2,
         & .423356164607534850D2,
         & .453801388984769080D2,
         & .484711813518352239D2,
         & .516066755677643736D2,
         & .547847293981123192D2,
         & .580036052229805199D2,
         & .612617017610020020D2,
         & .645575386270063311D2,
```

```
      & .678897431371815350D2,
      & .712570389671680090D2,
      & .746582363488301643D2
      &/
        DPQ = DBLE(IP)
        IF(IP.LE.30) STIRL = LNFACT(IP)
        IF(IP.GT.30) STIRL = ( DPQ + .5 ) * DLOG(DPQ) -
      & DPQ + K1 + K2/DPQ - K2/(.3D2 * DPQ * DPQ * DPQ)
        RETURN
        END
C-----------------------------------------------------------------------
        SUBROUTINE POISSY(P,R,E,PO,ISTAR,IFIN)
           DOUBLE PRECISION P,R,E,PO
             DO 4040  II = ISTAR+1,IFIN
               R = R - DEXP(P)
               E = E - R
 4040          P = P - DLOG(DBLE(II)) + PO
               IF (E .LE. 0.000001) E = 0.0
               IF ( R .LE. 0.0 ) E = 0.0
        RETURN
        END
C-----------------------------------------------------------------------
        SUBROUTINE SIZOP(I)
        COMMON/AVA/AVLOG(70),CRANK(70),IPS(70)
        COMMON/BASE/BEBO
        COMMON/NEXT/BBOLIB,DBOLIB,PROLIB,DRBLIB
        COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
        COMMON/PRICE/ICOST(70),NBEST(70)
         REAL BEBO(70)
         DOUBLE PRECISION BBOLIB(18),DBOLIB(18),PROLIB(18),DRBLIB(18)
         REAL QRANK(18)
C
C*****DETERMINE THE RANKING VALUE OF EACH PERMISSIBLE PURCHASE
C
       DO 4545 N = NOB - MAXRED, NOB
C
C      *****THE RANKING VALUE IS THE RATIO OF EBO REDUCTION TO COST
C           IF AVAILABILITY IS UNDEFINED.
C
          IF ( MODE .EQ. 1 ) QRANK(N) = (BEBO(I) - BBOLIB(N))/DBLE(N)
C
C      *****THE RANKING VALUE IS THE RATIO OF THE LOG OF AVAILABILITY
C           INCREASE TO COST, IF AVAILABILITY IS DEFINED.
C
 4545    IF ( MODE .EQ. 2 )
      &  QRANK(N) = ( DBLE(IPS(I)) * DLOG( 1.0D0 - BBOLIB(N)/
      &             (DBLE(IPS(I))*DBLE(SPB))) - AVLOG(I))/
      &             (DBLE(N) * DBLE(ICOST(I)))
C
        IF ( MODE .EQ. 1 ) QRANK(NOB+1) = ( BEBO(I) - BBOLIB(NOB+1) )
        IF ( MODE .EQ. 2 )
      & QRANK(NOB+1) = ( DBLE(IPS(I)) * DLOG( 1.0D0 - BBOLIB(NOB+1)/
      &                (DBLE(IPS(I))*DBLE(SPB))) - AVLOG(I))/
      &                 DBLE(ICOST(I))
```

```
C
C*****NOW CHOOSE THE MOST COST EFFECTIVE PURCHASE.
C
        CRANK(I) = QRANK(NOB+1)
        NBEST(I) = NOB + 1
        DO 5545 N = NOB - MAXRED, NOB + 1
           IF( QRANK(N) .LE. CRANK(I) ) GO TO 5545
           CRANK(I) = QRANK(N)
           NBEST(I) = N
 5545    CONTINUE
        RETURN
        END
C------------------------------------------------------------------------------ ----------
        SUBROUTINE COMPOP(IMAX)
        COMMON/AVA/AVLOG(70),CRANK(70),IPS(70)
        COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
         REAL TEMP(70)
         INTEGER IPOS(70)
C
C*****IF THIS IS THE FIRST RANKING OF THE RUN, RANK ALL OF THE
C      COMPONENTS.
C
       IF ( IDONE .EQ. 1 ) GO TO 5550
C
C      *****TEMPORARILY STORE THE COMPONENT RANKING VECTOR IN TEMP.
C
            DO 5150 L = 1,NCOMP
 5150          TEMP(L) = CRANK(L)
C
C*****RANK EACH COMPONENT.  ONCE IT IS RANKED, SET ITS TEMPORARY
C      RANKING VALUE TO ZERO.
C
       DO 5350 M = 1,NCOMP
C
C      *****HYPOTHESIS: COMPONENT 1 SHOULD BE RANKED IN POSITION M.
C
          IPOS(M) = 1
C
          DO 5250 L = 1,NCOMP
C
C      *****TEST HYPOTHESIS:  IS ANY UNRANKED COMPONENT MORE COST
C            EFFECTIVE THAN THE ONE CURRENTLY RANKED IN POSITION M?
C            IF SO, ASSIGN THE MORE EFFECTIVE COMPONENT TO M.
C
 5250        IF ( TEMP(L) .GT. TEMP(IPOS(M)) ) IPOS(M) = L
C
 5350     TEMP(IPOS(M)) = 0.0
C
       GO TO 5850
C*****UPDATE RANKINGS TO REFLECT THE LATEST PURCHASE
C
C
C*****FIND THE NEW RANKING POSITION OF THE MOST RECENTLY PURCHASED CO
C
```

```
 5550 NEWPOS = 1
 5650 NN = NEWPOS + 1
C
C     *****SEE IF THIS VALUE OF NEWPOS IS THE NEW RANKING POSITION
C          IF IMAX.
C
           IF ( CRANK(IMAX) .GE. CRANK(IPOS(NN)) ) GO TO 5750
C
C     *****IF IT IS NOT, THEN THE COMPONENT RANKED "NN" SHOULD BE
C          BUMPED UP ONE POSITION.
C
           IPOS(NEWPOS) = IPOS(NN)
           NEWPOS = NN
           IF ( NN .LT. NCOMP ) GO TO 5650
C
C     *****IF IT IS, THEN RANK IT.
C
 5750      IPOS(NEWPOS) = IMAX
C
 5850 IMAX = IPOS(1)
      RETURN
      END
C-----------------------------------------------------------------------
       SUBROUTINE UPDATE(I)
        COMMON/EBON/BEBON(70),DEBON(70)
       COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
       COMMON/PRICE/ICOST(70),NBEST(70)
       COMMON/SPARE/JSPARE(70),KSPARE(70)
       COMMON/SYST/SYSLOG,SYSTAV,TOTCST
C
C*****UPDATE THE SPARES INVENTORIES AND THE TOTAL COST COUNTER.
C
        BEBO(I) = BEBON(I)
        DEBO(I) = DEBON(I)
       JSPARE(I) = JSPARE(I) - NOB + NBEST(I)
       IF( NBEST(I) .EQ. NOB + 1 ) GO TO 4455
         KSPARE(I) = KSPARE(I) + 1
         TOTCST = TOTCST + NBEST(I) * ICOST(I)
         GO TO 5555
 4455  TOTCST = TOTCST + ICOST(I)
C
C*****UPDATE THE PURCHASED COMPONENT'S AVAILABILITY.  ALSO UPDATE
C      THE SYSTEM'S AVAILABILITY IF IT IS COMPUTABLE.
C
 5555  CALL AVAIL(I)
       RETURN
       END
C-----------------------------------------------------------------------
       SUBROUTINE AVAIL(I)
       COMMON/AVA/AVLOG(70),CRANK(70),IPS(70)
       COMMON/BASE/BEBO
       COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
       COMMON/PRICE/ICOST(70),NBEST(70)
       COMMON/SYST/SYSLOG,SYSTAV,TOTCST
```

```
      REAL BEBO(70)
C
C*****COMPUTE THE LN OF THE COMPONENT AVAILABILITY.  IF THE COMP.
C     HAS ALREADY BEEN RANKED, USE THE QUICK METHOD OF COMPUTA-
C     TION, STATEMENT NUMBER 6660.
C
       IF ( IDONE .EQ. 1 .OR. MODE .EQ. 2 ) GO TO 6660
       IF ( BEBO(I) .GE. FLOAT(IPS(I)) * SPB ) GO TO 9960
          AVLOG(I) = DBLE(IPS(I)) * DLOG( 1.0D0 - BEBO(I)/
     &                    (DBLE(IPS(I))*DBLE(SPB)))
          SYSLOG = SYSLOG + AVLOG(I)
          MODE = 2
          GO TO 9960
 6660  IF ( NBEST(I) .LE. NOB )
     & DELTA = DBLE(NBEST(I)) * DBLE(ICOST(I)) * DBLE(CRANK(I))
       IF ( NBEST(I) .EQ. NOB +.1 )
     & DELTA = DBLE(ICOST(I)) * DBLE(CRANK(I))
          AVLOG(I) = AVLOG(I) + DELTA
          SYSLOG = SYSLOG + DELTA
       IF ( SYSLOG .GE. -81.0D0 ) SYSTAV = EXP(SYSLOG)
 9960     CONTINUE
       RETURN
       END
C-------------------------------------------------------------
      SUBROUTINE PRELIM(I)
       COMMON/BASE/BEBO
       COMMON/DEPOT/DEBO,DMLOG,DPROB,DREBO
       COMMON/MEAN/BMEANO(70),DMEAN(70)
       COMMON/OPTS/IPRE
       COMMON/PARAMS/IDONE,MAXRED,MODE,NCOMP,NOB,SPB
       COMMON/PRICE/ICOST(70),NBEST(70)
       COMMON/SPARE/JSPARE(70),KSPARE(70)
       COMMON/SYST/SYSLOG,SYSTAV,TOTCST
        DOUBLE PRECISION AM,PO,P,R,U
        REAL BEBO(70)
        REAL DEBO(70),DMLOG(70),DPROB(70),DREBO(70)
C
C*****MAKE PRELIMINARY BUYS OF EACH COMPONENT FOR THE DEPOT
C     THE BASES, IF NECESSARY.
C
C
C     *****TEST: DOES THE DEPOT NEED A BUY?
C
          IF ( ( IPRE .EQ. 1 .AND. DMEAN(I) .LE. 10.0) .OR.
     &           ( IPRE .EQ. 2 .AND. DMEAN(I) .LE. 38.0) )
     &    GO TO 1005
C
C     *****IF IT DOES, SET THE DEBO VARIABLES AND BUY.*****
C
          P = DBLE(DPROB(I))
          R = DBLE(DREBO(I))
          E = DBLE(DEBO(I))
          PO = DBLE(DMLOG(I))
          AM = DBLE(DMEAN(I))
```

```
          LLL = IFIX(DMEAN(I))
C
C     *****IF USER OPTED FOR 2, BUY A PIPELINE'S WORTH.
C
          IF( IPRE .EQ. 2 ) CALL LIB(P,R,E,PO,AM,JSPARE(I))
C
C     *****IF USER OPTED FOR 1, BUY SIX STANDARD DEVIATIONS
C           LESS THAN A PIPELINE'S WORTH.
C
          IF( IPRE .EQ. 1 ) CALL CON(P,E,PO,AM,JSPARE(I),LLL)
C
C          *****ACCOUNT FOR THE GLUMP AT THE DEPOT
C
            DPROB(I) = P
            DREBO(I) = R
            DEBO(I) = E
            TOTCST = TOTCST + JSPARE(I) * ICOST(I)
C
C          *****COMPUTE BMEAN GIVEN THE PURCHASED SPARES.*****
C
            AM = DBLE(BMEANO(I)) - (AM-E)/DBLE(NOB)
            E = AM
C
C     *****TEST: DOES THE BASE NEED A BUY?
C
          IF ( ( IPRE .EQ. 2 .AND. AM .LE. 10.0 ) .OR.
     &          ( IPRE .EQ. 1 .AND. AM .LE. 38.0 ) )
     &    GO TO 1015
C
C     *****IF IT DOES, SET THE BEBO VARIABLES AND BUY.*****
C
          PO = DLOG(AM)
          LLL = IFIX(AM)
C
C     *****IF THE USER OPTED FOR 2, BUY A PIPLINE'S WORTH.
C
          IF ( IPRE .EQ. 2 ) CALL LIB(P,R,E,PO,AM,KSPARE(I))
C
C     *****IF THE USER OPTED FOR 1, BUY SIX STANDARD DEVIATIONS LESS
C           THAN A PIPELINE'S WORTH.
C
          IF ( IPRE .EQ. 1 ) CALL CON(P,E,PO,AM,KSPARE(I),LLL)
C
C*****UPDATE BEBO AND COST*****
C
 1015       BEBO(I) = E
            TOTCST = TOTCST + KSPARE(I) * ICOST(I) * NOB
 1005     CONTINUE
        RETURN
        END
```

*
TIME OUT

EPILOG

list

```
C*****THIS PROGRAM PROCESSES DRAMA OUTPUT.*****
          INTEGER KSPARE(70),JSPARE(70)
          INTEGER IPS(70),ICOST(70)
          IF ( NAVAIL .EQ. 0 ) NTABLE = 27
          IF ( NAVAIL .GT. 0 ) NTABLE = NAVAIL
          REAL BEBO(70),DEBO(70)
          CALL FMEDIA(9,0)
          CALL FMEDIA(10,0)
C          CALL FMEDIA(11,0)
C          CALL FMEDIA(12,0)
C          CALL FMEDIA(13,0)
C
C*****USER OPTION VARIABLES.*****
          IOUT = 1
          NAVAIL = 0
C
C*****INPUT SCENARIO DATA.******
          READ(9,109) NOB,SPB,NCOMP
C         *****INPUT COMPONENT DATA IF IOUT = 3.*****
          IF ( IOUT .LE. 2 ) GO TO 1000
          DO 999 I = 1,NCOMP
  999     READ(9,109) IPS(I),ICOST(I),BDUMMY,DDUMMY
  109     FORMAT(V)
 1000     CONTINUE
C
C*****PRINT AN OUTPUT TABLE FOR EACH AVAILABILITY SELECTED BY TH
C         USER
C
          DO 8000 IT = 1,NTABLE
            READ(10) KSPARE
          IF ( IOUT .GT. 1 ) READ(11)JSPARE
          IF ( IOUT .GT. 2 ) READ(12) BEBO
          IF ( IOUT .GT. 2 ) READ(13) DEBO
          IF ( NAVAIL .LT. NTABLE .AND. NAVAIL .GT. 0 ) GO TO 8000
C
C         *****INITIALIZE COUNTERS IF IOUT = 3.*****
          IF ( IOUT .LT. 3 ) GO TO 2000
          TCOST = 0.0
          DCOST = 0.0
          BCOST = 0.0
          DBOTOT = 0.0
          BBOTOT = 0.0
          AVTOT = 1.0
 2000     CONTINUE
C
C         *****PRINT A TABLE HEADER.*****
          IF ( IOUT .EQ. 1 ) WRITE (6,8016) IT
          IF ( IOUT .EQ. 2 ) WRITE(6,8026) IT
          IF ( IOUT .GT. 2 ) WRITE(6,8036) IT
 8016     FORMAT(1H1,20('*'),'BASE STOCKAGE LEVELS FOR AVAILABILITY
     &    ' LEVEL ',I3,11('*'),//)
 8026     FORMAT(1H1,10('*'),'TOTAL STOCKAGE LEVEL FOR AVAILABILITY
```

```
     & ' LEVEL ',I3,10('*'),//)
8036 FORMAT(1H1,10('*'),'INVENTORY AND BACKORDER QUANTITIES FOR',
     & ' AVAILABILITY LEVEL ',I3,10('*'),//)
C
C
C*****OUTPUT DATA TO TERMINAL.  PERFORM NECESSARY COMPUTATIONS
C       IF OPTION 3 OR 4 HAS BEEN SELECTED.
      DO 7000 I = 1,NCOMP
      ITAB = 0
C      *****START A NEW PAGE EVERY 40 LINES.*****
      IF( MOD(10,I) .EQ. 1 ) ITAB = 1
      IF ( ITAB .EQ. 1 .AND. I .GT. 1 ) WRITE(6,7006)
7006  FORMAT(1H1)
        IF ( IOUT .EQ. 2 ) GO TO 6200
        IF ( IOUT .GT. 2 ) GO TO 6300
C      *****OUTPUT FOR OPTION 1.*****
            IF ( ITAB .EQ. 1 ) WRITE(6,7016)
7016  FORMAT(' ','COMPONENT TYPE',5X,'BASE STOCK',//)
      WRITE(6,7017) I, KSPARE(I)
7017  FORMAT(' ',11X,I3,11X,I4)
      GO TO 7000
C
C      *****OUTPUT FOR OPTION TWO.*****
6200  IF ( ITAB .EQ. 1) WRITE(6,7026)
      WRITE( 6, 7027 ) I,JSPARE(I),KSPARE(I)
7026  FORMAT(' ','COMPONENT TYPE',5X,'DEPOT STOCK',5X,'BASE',
     & ' STOCK',//)
7027  FORMAT(' ',11X,I3,12X,I4,11X,I4)
      GO TO 7000
C
     *****OUTPUT FOR OPTIONS 3 AND 4*****
6300  CONTINUE
C          *****COMPUTE COSTS AND AVAILABILITIES*****
      LCOST = ( NOB*KSPARE(I) + JSPARE(I) ) * ICOST(I)
      TCOST = TCOST + LCOST
      DCOST = DCOST + JSPARE(I) * ICOST(I)
      BCOST = BCOST + (NOB*KSPARE(I)*ICOST(I))
      AVCOMP = ( 1.0 - BEBO(I) / ( FLOAT(IPS(I)) * SPB ) )
     & ** IPS(I)
      AVTOT = AVTOT * AVCOMP
      BBOTOT = BBOTOT + BEBO(I)
      DBOTOT = DBOTOT + DEBO(I)
C          *****OUTPUT COMPONENT DATA*****
      IF ( ITAB .EQ. 1 ) WRITE(6,7036)
      WRITE( 6,7037 )
     & I,JSPARE(I),KSPARE(I),LCOST,DEBO(I),BEBO(I),AVCOMP
7036  FORMAT(' ','COMP. TYPE',3X,'DEPOT STOCK',3X,'BASE STOCK',
     & 3X,'BASE EBO',3X,'DEPOT EBO',3X,'BASE EBO',3X,
     & 'AVAILABILITY',//)
7037  FORMAT(' ',7X,I3,10X,I4,9X,I5,3X,F9.3,2X,F9.3,5X,F10.6)
7000 CONTINUE
C
C
      IF ( IOUT .LT. 3 ) GO TO 8000
```

```
C
C*****WRITE A SUMMARY OUTPUT TABLE IF ONE IS CALLED FOR.*****
C
          WRITE(6,9006) IT
     9006 FORMAT(' ',1H1,'END ITEM SUMMARY FOR AVAILABILITY LEVEL ',
         & I3,///)
          WRITE(6,9016) DCOST
          WRITE(6,9026) BCOST
          WRITE(6,9036) TCOST
          WRITE(6,9046) DBOTOT
          WRITE(6,9056) BBOTOT
          WRITE(6,9066) AVTOT
     9016 FORMAT(' ','TOTAL COST OF DEPOT SPARES',24X,F10.0)
     9026 FORMAT(' ','TOTAL COST OF BASE SPARES',25X,F10.0)
     9036 FORMAT(' ','GRAND TOTAL COST',34X,F10.0)
     9046 FORMAT(' ','REMAINING EBO AT DEPOT',24X,F8.3)
     9056 FORMAT(' ','REMAINING EBO AT EACH BASE',20X,F8.3)
     9066 FORMAT(' ','END-ITEM AVAILABILITY',26X,F10.6)
     8000 CONTINUE
          STOP
          END

*
```

APPENDIX C

EXECUTING JOBS ON THE HONEYWELL 635

TIME-SHARING MODE

When DRAMA is being run for a small data base (250 items or less) with moderate demand (an average mean base demand across all components of 10 or less) and for a small number of bases (15 or less), it is practicable to run the model in a time-sharing mode. Setting up a time-sharing run is quite simple, for all that needs to be done (besides setting up the input data base with PROLOG) is to "attach" the input and output files to the main program. The easiest way to do this is to add a line of code to the main program for each file that needs to be attached. For example, the code required to attach the input file XM-1DATA is

```
CALL ATTACH(9,"OS29/N232D/XM-1DATA;", 3,0,ISTAT,)
```

The 9 in the parentheses is the logical file number for DRAMA inputs. The string OS29 is a catalog string (which differs among using organizations). The string N232D is a project-number string (which differs among users within a using organization).

If one wishes to see additional outputs he must attach the appropriate output files. The appropriate logical file number for each type of output file is designated below:

File Type	Logical File Number
Base Spares	10
Depot Spares	11
BEBO	12
DEBO	13

Therefore, if one wishes to output the Base Spares data to a file called BSPARES, he must add the following statement:

```
CALL ATTACH(9,"OS29/N232D/BSPARES;", 3,0, ISTAT,)
```

RUNNING IN REMOTE BATCH MODE

Jobs too large for time-sharing can usually be run in remote batch mode. To run in remote batch the user must: (1) put the DRAMA program into a form submittable in remote batch, (2) properly define the input and output files in a JCL program that will submit DRAMA in remote batch, and (3) establish suitable limits in the JCL program for DRAMA's running time, core requirement, and output requirement.

To put DRAMA into a form that can be submitted in remote batch, the user must change all CALL ATTACH statements in the program to CALL FMEDIA statements. For example, he must change the statement that attaches the input file, XM-1DATA, from

```
CALL ATTACH (9,"OS29/N232D/XM-1DATA;",3,0,ISTAT,)
```

to

```
CALL FMEDIA (9,0).
```

Similarly, he must change the statement that attaches the output file BSPARES from

```
CALL ATTACH (10,"OS29/N232D/BSPARES;",3,0,ISTAT,)
```

to

```
CALL FMEDIA (10,0).
```

To make sure that the correct file is input as logical file 9 (and to make sure that the correct files are output as logical files 10, 11, 12 and 13), the user must include a PERMFILE or PRMFL statement in his JCL. For example, to input the file XM-1DATA, the user should employ the statement

```
$:PRMFL:09,R,S,OS29/N232D/XM-1DATA
```

In this statement, 09 is the logical file number that will represent the file XM-1DATA in the main program, the letter R indicates that this file will be read, the letter S indicates that it is sequential, and the string OS29/N232D/XM-1DATA is the complete catalog/project number/file name of the input file.

To assign the output file "BSPARES" to the correct logical file number, the following is required:

$: PRMFL:10,W,S,OS79/N232D/BSPARES

The number 10 here stands for the logical file number of the output file, and the letter W indicates that this logical file must be written to, not read. For each logical file in the main program, there must be a PERMFILE statement in the Job Control Language program.

LIMITS FOR REMOTE BATCH JOBS

The limits for a remote batch job will need to be altered with the size of the job. The running-time limit of the job is driven by the number of components and by the size of BMEANØ and DMEAN. For a data base of 200 components in which the average value of BMEANØ was less than 10, 20 CPU seconds were found to be an adequate limit. When average demand is higher--say 25--then a 100-CPU-seconds limit would be more reasonable.

Core requirements are driven primarily by the number of components. Data bases of 200 components or less should require no more than 15K of core. Larger data bases--say 1000 components--will probably require more than 30K of core.

Output requirements of DRAMA are minimal, so we have arbitrarily established a nominal number of lines--2K--as the DRAMA output limit.

If a job exceeds one of its limits it will terminate, and the user will have to resubmit it with altered limits in order to obtain a complete run.

The LIMITS statement of the Honeywell JCL looks like this:

LIMITS:20,32K,,2K

In this statement 20 is the time limit in CPU seconds, 32K is the core limit in 6-bit words, and 2K is the output limit in lines.

THE JCL PROGRAM

A sample JCL program for a complete DRAMA run is presented below:

```
10##N,J
20$:IDENT:OS2033N232D ,OS29UFABBRO
30$:LIMITS:20,20K,,4K
60$:OPTION:FORTRAN
70$:FORTY:NDECK
90$:SELECTA:OS29/N232D/DRAMA
95$:LIMITS:2,20K,,2K
100$:EXECUTE
110$:LIMITS:20,20K,,2K
120$:PRMFL:09,R,S,OS29/N232D/XM-1DATA
125$:PRMFL:10,W,S,OS29/N232D/BSPARES
140$:ENDJOB
```

The IDENT statement contains the user's complete catalog-string. The first LIMITS statement defines the limits of the entire job; the OPTION statement denotes the language of the submitted program; the FORTY statement instructs the FORTRAN compiler to read the stored program as if it were a deck of cards; the SELECTA statement gets DRAMA; the second LIMITS statement establishes the limits for the compilation of DRAMA; the last LIMITS statement establishes limits for DRAMA's execution; and the PRMFL statements define the logical input and output files.

RUNNING THE JCL PROGRAM

To run the JCL program, the user must enter the Honeywell's CARDIN subsystem. He does this by keying in "CARDIN" when the operating system asks him what subsystem he wishes to use.

Once in CARDIN, the user can run the program by putting the JCL program into his workspace and then typing RUN. The operating system will respond to

this command by typing out a job reference number, or SNUMB. This SNUMB must later be used to retrieve output.

The recommended way to retrieve output is to type in the command

JOUT "SNUMB".

after giving the program a reasonable time to execute. Then the user can inspect his output from a remote terminal by using the Honeywell Time-Sharing System/Batch Interface facility. Finally, to get hard copy of his output, he should type in the command

DIRECT ONLINE

which will cause the output to be printed out at the line printer in the Air Force Data Services Center. Output will usually be printed out at the center within a few hours of the DIRECT ONLINE command.